ANNO 2 - NUMERO 6 APRILE/MAGGIO 1989 L. 14.000 - Frs. 21.00

GRUPPO EDITORIALE JACKSON
AREA CONSUMER

PASSE-PARTOUT NELL'UNIVERSO DI AMIGA

CONTIENE AMIGA Disk Magazine

L' Amiga che ha incastrato ROGER RABBIT

Tombola

Deluxe Paint III?

Ancora Musica

Lo studio di registrazione midi

TOP GAME:

AMIGABASIC
ASSEMBLY
AMIGA-C
MODULA-2

Spedizione in abbonamento postale Gruppo III/70 Taxe percue (Tassa riscossa) Milano Ferr. Corr.

IN COLLABORAZIONE CON

PRESENTA
LA GRANDE ENCICLOPEDIA DI INFORMATICA PER RAGAZZI

- IMPARIAMO A PROGRAMMARE
- COME È FATTO E COME FUNZIONA
- GIOCHI, GIOCHI, GIOCHI
- TECNOLOGIA E APPLICAZIONI
- SI FA... NON SI FA

L'informatica ti appassiona? vorresti approfondirla per studiare, giocare, curiosare? Inizialmente, per potersi orientare, è preferibile una guida giovane ma rigorosa e completa, senza risultare difficile o noiosa, da leggere tutta d'un fiato, corredata da centinaia di illustrazioni a colori.

VINCI 30 FAVOLOSI COMPUTER AMSTRAD CPC 6128

Aut. Min. Rich.

Questa è LA GRANDE ENCICLOPEDIA DI INFORMATICA PER RAGAZZI. Potrai entrare nei segreti del personal computer: storia, termini, concetti fondamentali, hardware, funzionamento e tanti programmi software in linguaggio Basic. JACKSON, in collaborazione con AMSTRAD, ti svela i segreti dell'informatica in fascicoli settimanali, **nella tua edicola**, a sole lire 2.500, da rilegare in 2 splendidi volumi illustrati.
Ma non è tutto!
Assieme al grande poster del Basic a colori, sul primo fascicolo scoprirai come vincere uno splendido computer AMSTRAD CPC 6128.

Sganciati dall'ordinario, entra nello straordinario!

La rivista di questo mese ospita articoli piuttosto impegnativi, ma siamo certi che i nostri lettori non si fermeranno davanti agli ostacoli che tali trattazioni imporranno loro. Anzi ne verranno senz'altro stimolati sì da spronarli ad avventurarsi ancora lungo l'impervia, ma ricca di soddisfazioni, strada che porta alla scoperta del pianeta Amiga.

Unico raggio di luce in tante tenebre un gioco un po' datato, ma pur sempre amato da chi lo ricorda come un compagno mai stanco d'allietare molte serate in famiglia o tra amici; quando più spesso s'era propensi a spegnere la televisione. Per chi non l'avesse ancora capito, sto parlando della Tombola, gioco per noi implementato su Amiga dal valente collaboratore De Sabbata; che si è sbizzarrito e soprattutto, divertito fra fagioli e chicchi di riso.

Fantastico anche il TOP GAMLE del mese: Driller, un gioco mozza fiato di cui questo mese vi diamo la demo version.

Per gli appassionati di musica il duo Laus (padre e figlio), ci propone un articolo sugli studi MIDI e loro varie applicazioni con Amiga... naturalmente.

Un'ultima breve annotazione la voglio dedicare ai vari corsi - dedicati ai più diversi linguaggi - proposti dalla nostra testata e soprattutto al rapporto che "dovrebbe" intercorrere tra quest'ultima ed i propri lettori. Scrivo ciò perché mi piacerebbe conoscere - non solo a me, ma anche ai vari autori- e verificare l'utilità di quanto si sta facendo non solo tramite lettere di complimenti (fanno comunque sempre piacere), ma anche per mezzo di lavori fatti da chi questi corsi segue. Prego pertanto gli interesati ad inviarci dei loro elaborati. E non vi dovete preoccupare che si tratti di programmi eccezionali, basta che siano stati prodotti, appunto, grazie alle nozioni acquisite dai nostri corsi.

Nella speranza di ricevere presto una marea di materiale vi salutiamo.

Associato al

Testata aderente al C.S.S.T.
non soggetta a certificazione
obbligatoria in quanto la presenza
pubblicitaria è inferiore al 10%

Foto di copertina realizzata
nellla redazione grafica della:
Graphic & Comp di Gorizia

Anno II Numero 6 Aprile/Maggio 1989

DIRETTORE RESPONSABILE
Paolo Reina

REDAZIONE
Graphic & Comp. Gorizia

COORDINAMENTO REDAZIONALE
Simone Concina

ART DIRECTOR
Gianni Marega

COLLABORATORI
Marco Arpini
Alex Cozzi
Piero Dell'Oste
Avelino Desabbata
Giorgio Dose
Fabio Fasciani
Antonio Ferrillo
Aldo Laus
Andrea Laus
Furio Lusnig
Luigi Manzo
Giovanni Michelon
Emilio Orione
Alessandro Prandi
Giacomo Pueroni
Nicola Reale
Paolo Russo
Giuseppe Salmoiraghi

**GRAFICA
IMPAGINAZIONE, COPERTINA**
Graphic & Comp.

DIVISIONE PUBBLICITA'
Via Pola, 9 - 20124 MILANO - Tel. 69.481
Telex 316213 REINAI - 333436 GEJ - ITI
OVERSEAS DEPARTMENT: Tel. 02/6948201
PUBBLICITA' PER ROMA-LAZIO E CENTRO SUD
Via Lago di Tana, 16 - 00199 Roma
Tel. (06) 8380547 - Telefax (06) 8380637

STAMPA
Grafika, 78 - Pioltello

DISTRIBUZIONE
Sodip - Via Zuretti, 25 - 20125 MILANO
Spedizione in abbonamento postale Gruppo III/70
Pubblicità inferiore al 70%

UFFICIO ABBONAMENTI
Tel. (02) 6122527 - 6187376
Prezzo della rivista L. 14.000 (Frs. 21.00)
Numero arretrato L. 28.000
I versamenti vanno indirizzati a:
Gruppo Editoriale Jackson
Via Rosellini, 12 - 20124 Milano
mediante emissione di assegno bancario, vaglia o utilizzando il C/C postale numero 11666203 Per i cambi di indirizzo, indicare, oltre al nuovo, anche l'indirizzo precedente, ed allegare L. 500, anche in francobolli.

**DIREZIONE,
REDAZIONE, AMMINISTRAZIONE**
Via Rosellini, 12 - 20124 Milano
Tel. (02) 69481-6880951/2/3/4/5 - Telex 333436 GEJ - ITI

SEDE LEGALE
Via G. Pozzone, 5 - 20121 Milano

Il Gruppo Editoriale Jackson
è iscritto nel Registro nazionale della Stampa
al n. 117 vol. 2 - foglio 129 in data 17/8/1982

Autorizzazione del Tribunale di Milano n. 102
del 22/2/1988

Computer Lab

A Milano è stato aperto un nuovo Centro Assistenza Autorizzato per prodotti Commodore. L'azienda, denominata COMPUTER LAB, ha il proprio laboratorio sito nella zona del centro in viale Monte Nero 66. Il tempo massimo per la riparazione delle apparecchiature si aggira intorno ai 5 giorni per la gamma home (VIC-20, C-16, C-64, C-128, AMIGA 500) e intorno alle 48 ore per la gamma professional (MS-DOS, AMIGA 1000, AMIGA 2000, ecc.).

Software

IMPORTANTE!!! Le ultime novità immesse sul mercato dalla LEADER.

PRO SOUND DESIGNER AMIGA (£.199.000) è un software disponibile per tutti i modelli Amiga, eccellente per il campionamento a 4 canali e contiene alcune funzioni finora possedute solo da sistemi professionali. È possibile monitorare e intonare precisamente un campionamento mono o stereo. Durante il monitoraggio, si può sfruttare la visualizzazione della Forma d'Onda stereo per ottenere un livello corretto e un'alta qualità del suono che deve essere campionato.

PRO MIDI PLUS AMIGA (£.85.000) disponibile per tutti i modelli Amiga. Richiede l'interfaccia MIDI MM3000 o una simile. Con il Pro Midi Plus è possibile suonare 4 canali e salvare e caricare gruppi di strumenti. Collegando la tastiera MM5000 all'Amiga, potete creare un potente sintetizzatore e campionatore. È compatibile con i campionamenti salvati nel formato 85VX IFF, con la tastiera musicale MM5000 e con tutti gli strumenti MIDI.

INTERFACCIA M.I.D.I (£.95.000) è compatibile con tutti i programma MIDI per Amiga ed è dotata di standard MIDI IN, MIDI OUT, MIDI THROUGH.

AMIGA MUSIC SYSTEM (£.699.000): sistema musicale completo per trasformare l'Amiga in una tastiera, sintetizzatore, campionatore, interfaccia Midi.

Il sistema comprende:
- PRO SOUND DESIGNER
- PRO MIDI PLUS
- INTERFACCIA M.I.D.I.
- MM 5000; tastiera musicale a 5 ottave con i tasti di dimensioni normali e di alta qualità. È compatibile con tutti i modelli Amiga. Il software fornito con la tastiera consente l'utilizzo della MM 5000 come una tastiera MIDI. È inoltre compatibile con moltissimi programmi per Amiga incluso il Deluxe Music e quelli della serie Eidersoft Professional Music Systems.

ESPIONAGE: In questo game sarete al comando di dodici agenti scelti di una organizzazione internazionale di spionaggio. Gli agenti devono partire dalle loro basi segrete ed avanzare nello scenario costituito da città, aereoporti, deserti, ecc. di tutto il mondo.

Il destino della terra è ora nelle vostre mani in un gioco di violenti conflitti per ottenere i 4 microfilm contenenti importanti informazioni su un'arma di interesse mondiale.

BATMAN: Il gioco è ambientato nella città di Gotham City, e verrete coinvolti nelle vesti dell noto eroe dei fumetti che deve difendersi dai pericoli causati da The Penguin e The Joker. Attraverso varie difficoltà dovrete cercare di liberare il vostro partner Robin e cercare di terminare il gioco che è provvisto di un fantastico look da fumetto!

ROBOCOP: Ispirato dal film, lo scenario del gioco si svolge in un vicino futuro, dove la vecchia Detroit vive assediata dalla delinquenza. La giustizia è diventata un business, la Omni Consumer Products pensa di sostituire gli agenti con dei robot. L'esperimento fallisce ed il cervello del poliziotto Weller viene utilizzato per realizzare un cyborg dalle caratteristiche super-umane: ROBOCOP!

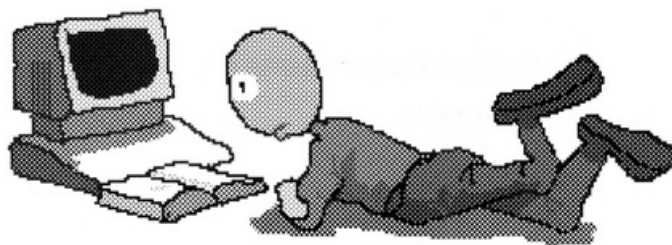
Efficiente ed implacabile, Robocop finisce con il diventare il paladino della vecchia Detroit.

MANHATTAN DEALERS: Indossate i panni dell'ispettore Harry e la vostra missione è di sequestrare e distruggere tutta la droga introdotta a Manhattan.

Per riuscirci dovrete sconfiggere l'organizzazione internazionale conosciuta come Manhattan Dealers.

Dovrete battere i vari avversari che troverete nelle zone di Harlem, sulle rive dell'Hudson, nei ghetti del Bronx e per le vie di Chinatown.

Sono inoltre disponibili per l'Amiga i seguenti programmi: Motor Massacre, Speedball, Mickey Mouse, Adv. Sky Simulator, Circus Games, Superman.

Questi programmi vengono distribuiti in Italia dalla LEADER s.r.l., via Mazzini 15, 21020 Casciago (VA).

Egregia Redazione di Amiga Magazine, vi scrivo per rammaricarmi dell'occasione persa dalla vostra rivista di diventare il più autorevole magazine italiano sul nostro beneamato computer. Aver chiamato il vostro periodico Amiga Magazine è stato fuorviante: meglio sarebbe stato intitolarlo "Lo zibaldone dei programmatori", mensile di accademia per addetti ai lavori che non si curano affatto di farsi capire dal resto del mondo.

Sorry, ma una unità del resto del mondo, e cioè il sottoscritto, si aspetta ben altro da una rivista su Amiga. Innanzitutto un linguaggio accessibile, tale per cui non sia obbligato a fermarmi già al primo articolo degli Amiga tricks alla parola "linguaggio (scusate il bisticcio) third party". Sempre sugli Amiga tricks: le notizie date erano caotiche, confuse ed imprecise. Ve ne siete accorti? E ancora, se volete fare un corso di basic (another brick in the wall!), prendete esempio da quello tenuto su Amiga World, non perdetevi in pedanti disquisizioni tecniche che hanno il solo effetto di fare passare immediatamente all'articolo successivo, e cioè dalla padella alla brace: informatica, algoritmi, un incubo di equazioni, ecc.

Gente, ma siete proprio sicuri che il vostro target di lettori sia interessato a cose del genere? I listati oramai non li sopporta più nessuno: sono sempre pieni di errori, ci vogliono tre notti insonni per impostarli e correggerli, e se poi sono anche compresi nel disco diventano doppiamente inutili. Il resto del dischetto lo tralascio: Reversi e Life il mio vetusto Apple II (anno 1976) me li aveva già mostrati almeno 10 anni orsono.

Insomma, volendo trarre una conclusione: avete preso come modello Amiga World in sedicesimo, ed avete aggiunto un dischetto, che oggi va di moda. Gli Amiga User italiani, dai contatti che ho avuto in questi 8 mesi di appartenenza alla setta, sono composti per il 70% da ragazzini indiavolati, affamati di novità, software e know-how intelligente. Del restante 30%, il 20% è fatto di gente come me. A tutti questi forse il modello imperfetto di Amiga User (la rivista) va abbastanza bene: tante novità e poca noia, specie dopo una giornataccia di lavoro. Il 10% finale è fatto di "spaccabit", persone cioè capaci di rompere un bit in quattro ma assai carenti nel rapporto con tutto ciò che non è computer (compresi i comuni mortali). Bene, la rivista mi sembra dedicata interamente a quest'ultima fetta di mercato.

Con questo voglio dire che il mix di utenza attuale, e il mondo di Amiga ne è un esempio esasperato, è ad anni luce dal periodo pionieristico che ha fatto la fortuna dei primi anni di "Bit" (cui peraltro AM un po' assomiglia): grandi listati, grandi disquisizioni tecniche, linguaggio da iniziati, pochi adepti, filosofia spicciola. Da una rivista su Amiga, oggi, non mi aspetto corsi di informatica o dissertazioni su Modula-2 (in libreria trovo di meglio, se ne ho voglia); mi aspetto invece di essere tenuto al corrente su tutte le novità (e per tali non intendo Test Drive) che riguardano il mio computer, mi aspetto di avere consigli intellegibili ed intelligenti che mi aiutino a districarmi nel sistema operativo e nel software applicativo, ed anche un po' di logica programmatoria (ma proprio poca e ben mirata): tutto il resto fa brodo, ma solo quello.

Ho finito. Scusate la letteraccia, normalmente non ne scrivo mai, ma non mi è mai capitato come in questo caso di trovarmi così fuori sintonia con una rivista. Spero comunque che sappiate trarre profitto dalle critiche, e risorgere come l'Araba Fenice: per cavalcare una tigre come Amiga ci vuole questo ed altro. Saluti e baci.

Danilo Lamera
Milano

Riporto questa lettera non certo per risponderle punto su punto, per giustificare delle scelte che , come le sue affermazioni, sarebbero comunque opinabili, ma semplicemente per renderla pubblica, per permettere a chi condivide tali sue impressioni di comunicarcele tramite questa rubrica.

Non so se quanto dice è tutto vero, probabilmente no, ma so con certezza che è mio preciso compito tenerne conto: nel caso che altre lettere dovessero seguire la sua.

Nel salutarla e ringraziarla per l'attenzione, spero abbia comunque notato che alcuni cambiamenti apportati alla testata vanno in parte proprio nella direzione da lei auspicata.

OBIETTIVO CHIAREZZA

La testata che state leggendo è iscritta al Consorzio e come tale è soggetta agli obblighi previsti dal regolamento.

Consorzio Stampa Specializzata Tecnica

Libera associazione di editori nata con l'obiettivo di tutelare il settore delle riviste specializzate e tecniche e di fornire dati certificati nonché informazioni chiare sulla loro diffusione e readership... per la più facile e sicura pianificazione pubblicitaria.

Consorzio Stampa Specializzata Tecnica

Via Pantano, 9
20121 MILANO
Tel. 02 - 8823298/235

Come usare Makedir da Basic

Dopo aver steso il vostro programma basic, probabilmente lo vorrete salvare in qualche directory del disco, se questa già esiste non c'è problema, ma viceversa, se la dovete creare dovrete andare in CLI ed eseguire il comando:

```
MakeDir df0:Prova
oppure
MakeDir df1:Prova/Esempio
```

Per evitare di uscire dall'AmigaBasic potete utilizzare questa routine che simula il comando MakeDir. Agendo sulla stringa DirName$ potete cambiare nome alla directory o creare addirittura una sottodirectory. Condizione inscindibile per il funzionamento di questa routine è la presenza del file Dos.BMap, il quale si trova nella directory BasicDemos del dischetto Extras.

```
DECLARE FUNCTION CreateDir& LIBRARY
DECLARE FUNCTION IoErr& LIBRARY
DECLARE FUNCTION UnLock& LIBRARY
LIBRARY "Dos.Library"
DirName$ = "Prova"
DirName$ = DirName$ + CHR$(0)
successo& = 0 THEN
IF successo& = 0 THEN
    errore& = IoErr(0&)
  PRINT "Numero Errore: ";errore&
 ELSE
  prova& = UnLock& (successo&)
END IF
LIBRARY CLOSE
```

Comando FileNote

Il comando FileNote è certamente tra i meno usati nella gestione dei file con Amiga, se provate però ad usarlo per un po' di tempo, noterete che esso è molto utile e immensamente pratico. Per prima cosa vediamo come si usa FileNote da CLI; per assegnare un commento a un file dovrete scrivere il comando con la seguente sintassi:

```
FILENOTE NomeFile COMMENT Prova
```

Per NomeFile si intende il file al quale vogliamo allegare la nota, nel caso dell'esempio sopra menzionato il commento al file sarà 'Prova'. Per visualizzare la scritta ' Prova ' dovete eseguire il List del dischetto o della directory dove il file è contenuto. Dopo avere svolto questa operazione potete andare al Workbench e clickare sull'icona del file e, quindi, selezionare la voce Info dal menu; a questo punto noterete che nello spazio riservato al commento c'è la scritta 'Prova', precedentemente impostate da CLI. Per cancellare un qualsiasi commento allegato a un file dovete ripetere l'istruzione precedentemente illustrata con l'unica differenza che dopo COMMENT dovete scrivere "".

```
FILENOTE NomeFile COMMENT ""
```

Ricordate che se copiate un file, l'istruzione Copy non legge la riga di commento per cui il nuovo file ne sarà privo. Un'altro importante aspetto sta nel fatto che potendo il commento raggiungere ben 80 caratteri ovviamente diventa indispensabile far uso degli spazi (per esempio: "Questa è una prova") in tal caso ricordate di inserire la frase tra virgolette.

```
FILENOTE NomeFile COMMENT "Questa è una prova"
```

Ora esaminiamo lo stesso comando usando il Workbench. Unica prerogativa per poter adoperare il comando FileNote è che il file sia rappresentato da un'icona nello schermo Workbench. Selezionando l'icona e quindi scegliendo l'istruzione Info dal menu verrà visualizzata una schermata contenente tutte le informazioni relative al file. A questo punto possiamo inserire un commento di 80 caratteri nel gadget COMMENT, diversamente dal CLI qui possiamo omettere le virgolette pur usando gli spazi. Ricordate che se inserite un commento o ne modificate uno già esistente prima di uscire dalla schermata Info dovete selezionare il gadget SAVE in basso a sinistra. Se volete cancellare una scritta già esistente basterà clickare con il tasto sinistro del mouse sul gadget COMMENT e quindi premere il tasto destro Amiga e il tasto 'X'. Se volete cancellare un commento a questo punto sarà sufficiente selezionare SAVE in basso a sinistra.

Più spazio sul disco Workbench

Ecco un semplice consiglio per recuperare spazio sul vostro disco Workbench. Sicuramente molti utenti Amiga hanno personalizzato e ottimizzato il loro principale strumento di lavoro ovvero il disco Workbench. Dopo aver modificato la Startup-Sequence nella directory 's' e aver scelto le varie opzioni possibili in Preferences potrete quindi pensare a come svuotare il dischetto delle parti superflue.

Innanzitutto dovete cambiare il nome al nuovo disco Workbench e usare il disco originale come boot-disk. A questo punto, per esempio, potete cancellare l'intera directory Fonts dal 'vostro' dischetto usando il comando 'Delete Fonts All', se date ora l'istruzione info da CLI noterete che dalla superficie del disco occupata al 99% siete passati all'87%. Se non possedete la scheda Janus potete

tranquillamente cancellare la directory PC (Delete PC All), ora siamo al 76%. Assieme alla directory PC potete quindi cancellare le altre directory come Sidecar e Expansion. Ovviamente ci sono anche dei file come DJMount nella directory 'c', che possono essere omessi dal disco nel caso non ci si serva della Janus, inoltre, potete penetrare nella sottodirectory 'Keymaps' della directory 'devs' e togliere le varie configurazioni di tastiera (usa0, usa1 eccetera), lasciando ovviamente il file 'i'. Dopo aver cancellato la directory Utilities eliminate anche il file Preferences, infine togliete tutti i file '.info' relativi ai file e directory cancellate. Se ora ridate il comando 'info' potrete notare che il vostro dischetto è occupato solo al 55%. Questo disco può adesso ospitare l'AmigaBasic e un wordprocessor come Textcraft e ancora vi resterebbe dello spazio per le vostre utility.

Nomi delle subroutine in Basic

Un buon metodo da seguire nella programmazione in Basic su Amiga è quello di nominare le varie subroutine definendone chiaramente il loro compito, per esempio:

```
CALCOLOANGOLOINGRADI:
DIVISIONENUMEROINPUT:
```

Ovviamente come si può ben intuire è immensamente scomodo durante la stesura del programma listare le varie subroutine in quanto bisogna scrivere list e il nome della routine, per aggirare questo ostacolo si possono alfabetizzare le varie subroutine, per esempio:

```
C: 'CALCOLOANGOLOINGRADI
D: 'DIVISIONENUMEROINPUT
```

Naturalmente è comunque utile prender nota su carta delle varie combinazioni tra alfabeto e nomi. Dopo le difficoltà iniziali vedrete che lavorare con questo metodo è estremamente comodo.

Comando Search & Replace in Basic

Per quanto non ci si possa lamentare eccessivamente dell'AmigaBasic bisogna però denunciare le sue carenze nell'editor. Infatti, se alla fine della stesura di un programma vi accorgete che dovete rinominare una certa variabile per un qualsiasi motivo, noterete appunto i limiti dell'editor. Non preoccupatevi troppo, salvate il programma basic come file ASCII, usando l'opzione ',A' del comando SAVE. Esempio:

```
SAVE "NomeProg.tes",A
```

Con il suffisso '.tes' si intende distinguere il file come file testo (ASCII) in modo da poterlo riconoscere successivamente. Ora potete usare un qualsiasi editor, un wordprocessor o l'ED del CLI. Se optate per quest'ultimo e però non avete molta dimestichezza con i comandi di ricerca (Find) allora premete il tasto 'Esc' e quindi digitate:

```
f/parola/
```

(per ' parola ' si intende il set di caratteri che si vuole sostituire) premete Return. dopo aver trovato il primo termine premete CTRL-G e passerete al secondo e così via. Ricordiamo che una lista dei comandi del Editor Amiga è stata pubblicata su Amiga Magazine n.1 (pagine 25-26).

Meno rumori

L'abilità dell'Amiga nel multitasking è certamente indubbia, però talvolta questo meraviglioso aspetto della macchina rischia di perdere in grazia quando il drive tenta il multitasking e la testina gracchia rumorosamente avanti e indietro sul dischetto. Ciò si verifica in modo particolare nei primi modelli Amiga. Per esempio durante l'esecuzione della Startup-Sequence se avete le istruzioni:

```
LOADWB
SHELL
```

disposte in questo modo noterete che prima che finisca il caricamento del Workbench il drive incomincia a gracchiare nel tentativo di caricare contemporaneamente il Shell. Per evitare questi sgradevoli gracidii, potete usare il comando Wait come segue:

```
LOADWB
WAIT 5
SHELL
```

Non crediate che facendo passare cinque secondi tra LOADWB e SHELL si perda del tempo, in quanto il computer evita di caricare inutilmente Shell durante il caricamento del Workbench.

Titoli in Basic

Una scritta scorrevole all'inizio dell'esecuzione di un programma certamente rende lo stesso molto più professionale e interessante, inoltre perché non approfittare della semplicità del linguaggio Basic?

```
REM Presentazione:
REM DIM SHARED SL%(6+(y2%-y1%+1)*2*INT((ABS(DeltaX)+16)/16)*Grandezza)
REM   x1%;y1% , x2%,y2% : Campo dello Scroll
REM   Numero        : Frequenza dello Scroll
REM   DeltaX        : Variabile DeltaX

SUB Scrittura(x1%,y1%,x2%,y2%,Contatore,DeltaX) STATIC
  DX=(ABS(DeltaX)-1)*SGN(DeltaX)
  FOR a=1 TO Contatore
   IF DeltaX<0 THEN GET(x1%,y1%)-(x1%-DX,y2%),SL% :ELSE GET(x2%,y1%)-(x2%-DX,y2%),SL%
    SCROLL (x1%,y1%)-(x2%,y2%),DeltaX,0
   IF DeltaX<0 THEN PUT(x2%+DX,y1%),SL%,PSET :ELSE PUT (x1%,y1%),SL%,PSET
  NEXT
END SUB
```

La chiamata alla subroutine suona così:

```
CALL Scrittura(x1%,y1%,x2%,y2%,Contatore,DeltaX)
```

Perché il sottoprogramma funzioni bisogna innanzitutto definire l'array LS%:

```
DIM SHARED SL%(6+(y2%-y1%+1)*2*INT((ABS(DeltaX)+16)/16)*Grandezza)
```

La profondità (grandezza) da il numero dei Bitplane. Ecco un esempio:

```
DIM SHARED SL%(6+(150-120+1)*2*INT((ABS(2)+16)/16)*16)
FOR a=1 TO 22:FOR a1=1 TO 5:PRINT " Amiga Magazine";
NEXT a1:PRINT:NEXT a:
CALL Scrittura (56,120,527,150,300!,2!) :END
```

Questo sottoprogramma lo potete migliorare e inserire nei vostri listati Basic usando il comando MERGE. Le migliorie da apportare a questa subroutine sono numerose e dipendono solo dalla vostra fantasia.

di Aldo e Andrea Laus

Questo articolo illustra le possibilità applicative del computer nel campo della registrazione musicale via sequencer MIDI.

Vengono descritti i componenti essenziali, le tecniche usate e le modalità per realizzare, tramite un programma sequencer, uno studio musicale di registrazione su nastro.

Grazie alla vasta gamma di componenti disponibili oggi sul mercato, è possibile iniziare l'approccio a questa tecnica anche con mezzi abbastanza modesti.

La qualità del prodotto finale, dipenderà ovviamente, oltre che dalla vostra abilità di "musicisti informatici", dalle qualità sia degli strumenti musicali impiegati che dalle caratteristiche del registratore e degli eventuali effetti, nonchè dal programma usato.

Per la serie: Amiga Workstation MIDI.

Lo studio di registrazione MIDI

Prima o poi, chi è appassionato di musica ed ha la fortuna di saper suonare uno strumento musicale, magari a tastiera, è attratto dal desiderio di suonare insieme a qualche suo simile o, in mancanza di questi, di incidere un accompagnamento che poi, in fase di ascolto, gli fornirà la base su cui suonare l'assolo.

Se siete musicisti e siete già arrivati a questo stadio e non disdegnate l'informatica, questo articolo è per voi.

Oggi sono disponibili moltissimi sintetizzatori musicali che generano suoni di strumenti con un realismo impressionante e che sono equipaggiati con decine di timbri.

Questi stessi strumenti sono sistematicamente dotati di un'interfaccia MIDI, ma, nonostante ciò non è ancora molto dif-

STUDIO DI REGISTRAZIONE MIDI

fusa la conoscenza di quali reali possibilità si possano ottenere dalla presenza di questo accessorio digitale che viene ostentato sotto forma di 2 o 3 prese poste generalmente sul retro dello strumento.

Forse tra le tante, la più interessante è la realizzazione di uno studio di registrazione MIDI.

Anche se l'idea può sembrare, a prima vista, molto ambiziosa, per iniziare ad ottenere apprezzabili risultati in questo senso non occorrono grandi cose comunque.

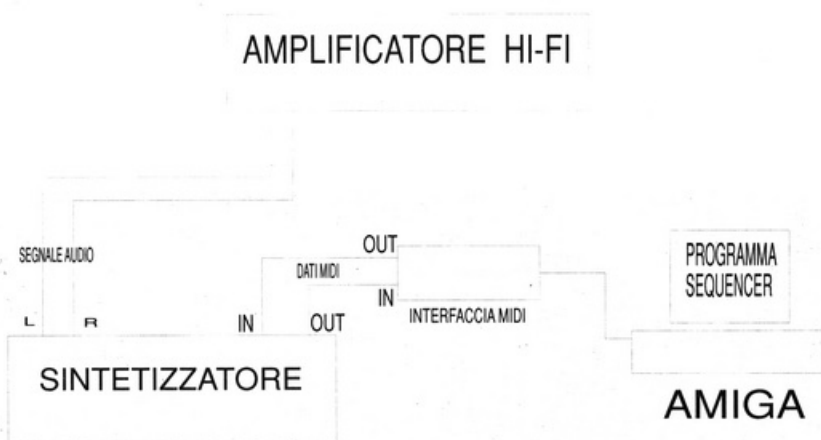
Con un computer Amiga, equipaggiato con una semplice interfaccia MIDI ed un opportuno programma (e cene sono già molti in giro), è possibile realizzare tramite il proprio sintetizzatore MIDI delle composizioni musicali polifoniche di grande realismo ed ottenere incisioni su nastro impeccabili e di qualità quasi professionale.

Questo dipende, come già detto prima dalla qualità intrinseca della attrezzatura usata.

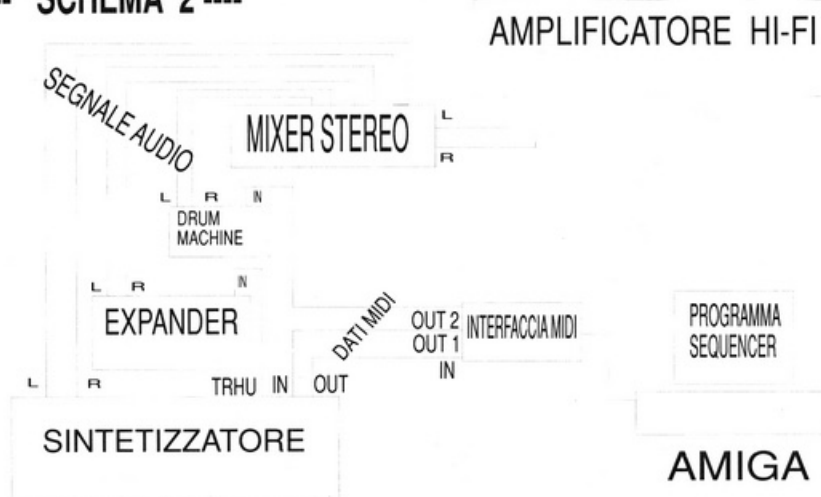
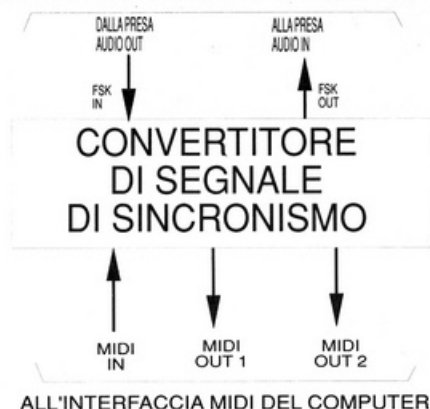
Registrazione tradizionale

Normalmente, l'appassionato musicale è attratto verso questi moderni sintetizzatori proprio dalle sonorità cristalline dei timbri e dagli effetti disponibili pensando ad un utilizzo in tempo reale dello strumento tramite la tastiera o per mezzo della varietà di controllori MIDI oggi disponibili.

E' naturale che, dopo aver gustato individualmente la qualità di tutti i suoni a disposizione, venga il desiderio di poter fare un'incisione su di uno di quei registratori multipista che oggi sono disponibili a prezzi abbastanza attraenti.

Il modo tradizionale di fare le registrazioni multipista consiste nell'eseguire le

diverse parti in tempo reale una dopo l'altra, sistemandole ciascuna su una pista, salvo effettuare sovrapposizioni successive da una traccia all'altra in modo da lasciarne una sempre libera onde poter rifare l'ultima registrazione, qualora ci siano stati errori o non si sia soddisfatti del risultato.

Vediamo ora alcuni dei problemi che si incontrano con questa tecnica:

- Se il numero di parti da incidere supera il numero di piste disponibili nel registratore, è necessario ricorrere a delle sovrapposizioni di piste e quindi avremo l'inconveniente del rumore di fondo (il famoso fruscio) che si accumula.
- Rigidità del sistema, in quanto, se dopo aver realizzato il lavoro, ci si rende conto che l'insieme timbrico non è quello che ci si aspettava, occorre rifare una o più piste (ancora peggio se esse sono già sovrapposte) con voci strumentali diverse e forse ripetere più volte l'esperimento fino a quando si è soddisfatti.
- Un'ulteriore rigidità è costituita dalla velocità di esecuzione: il registratore a nastro restituisce esattamente ciò che è stato eseguito e quindi, se riteniamo che il tempo del pezzo debba essere variato, ahimè, bisogna rifarlo.
- Un problema analogo sorge quando si vuole trasporre di tonalità il proprio pezzo, quando si vuole modificarne solo qualche nota, o quando si desidera ascoltare ciascuna parte separatamente, ma ormai le sovrapposizioni sono fatte.

Cosa serve

A risolvere brillantemente questi e altri problemi ci pensa il sistema MIDI.

Possiamo infatti sia liberarci dei fastidi visti sopra, che aprire inedite possibilità di manipolazione sul tessuto musicale da noi prodotto per mezzo di un sequencer MIDI che rappresenta il cuore dello studio di registrazione MIDI.

Accontentiamo subito i curiosi che vogliono sapere come si usa, cosa si ottiene e cosa serve per realizzare una registrazione con un sequencer MIDI.

Salvo smentite, dovute a novità dell'ultima ora (in elettronica oggi tutto è possibile), un musicista, che voglia "midizzarsi sequenzializzandosi tramite computer" (bah!), deve dotarsi del seguente equi-

paggiamento fondamentale:

A) Un computer (possibilmente dotato di una buona quantità di memoria, almeno 500 K).

B) Una interfaccia MIDI per il suddetto computer.

C) Uno o più strumenti musicali MIDI e relativa amplificazione (in uno studio casalingo può andar bene anche un impianto Hi-Fi con presa AUX).

D) Un programma sequencer

E) Un registratore a nastro, meglio se stereo (magari incorporato nell' Hi-Fi domestico) oppure un registratore multipista (ottimi anche quelli a cassetta) per riversarvi il risultato audio dei nostri sforzi.

Daremo più avanti che cosa determina la necessità di uno o dell'altro tipo.

Passiamo quindi ad analizzare le componenti essenziali del sistema che abbiamo enunciato, trascurando solo i punti A e B, in quanto, per il primo punto, l'Amiga, già nella configurazione 500 si presta ottimamente a supportare applicazioni MIDI, grazie anche ai numerosi programmi oggi disponibili.

Per quanto riguarda il punto B, dobbiamo dire solo che, sul mercato, sono disponibili diverse proposte per l'interfaccia MIDI per Amiga.

Per chi fosse interessato ad ulteriori dettagli, ricordiamo che, sul numero 5 di questa rivista, abbiamo dedicato un articolo all'analisi di questa importante componente della workstation MIDI.

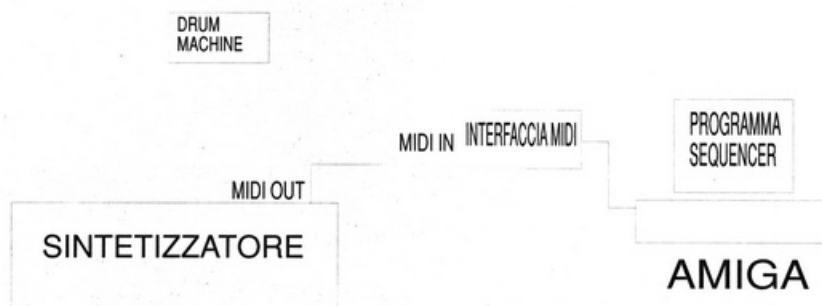
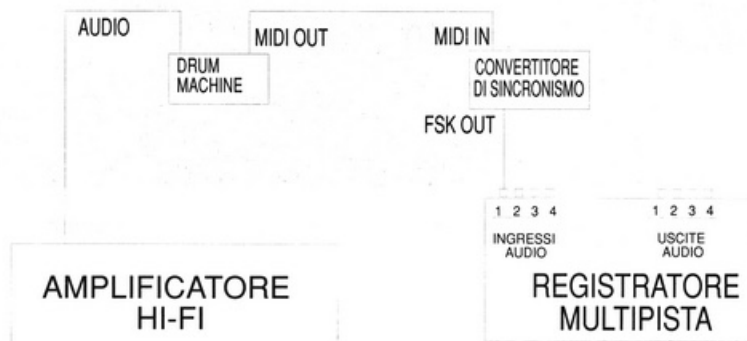
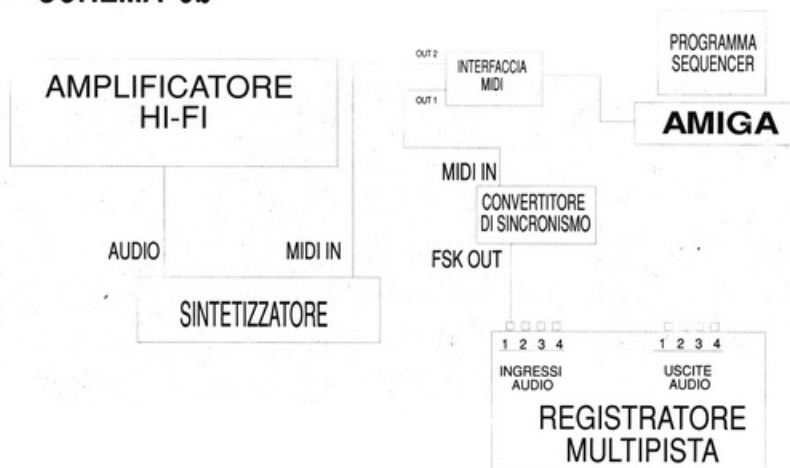
Proseguiamo quindi con gli altri componenti del sistema.

Generatori di suono

L'aspetto forse più critico o per lo meno sul quale occorre avere le idee più chiare è il tipo di generatore di suoni più o meno adatto allo scopo.

Non vogliamo in questa sede entrare nel merito del tipo di tecnica di generazione (analogica, digitale, pcm ecc.), bensì vogliamo chiarire, speriamo, le possibilità di utilizzazione per registrazioni MIDI polifoniche dei vari tipi di strumenti oggi reperibili sul mercato.

Quindi, semplificando la classificazione assimiliamo le varie tipologie di strumenti, sintetizzatori, expanders ecc. al termine generatori di suono.

Facciamo un altro passo e diciamo che ogni generatore di suono è costituito da

Dopo l'avvento del MIDI il "posto di lavoro" del musicista è sempre più simile ad una workstation informatica

un certo numero di voci, che sono poi quelle che producono il suono di una singola nota.

Il numero di voci determina la polifonia dello strumento ovvero il numero massimo di note che possono essere suonate insieme.

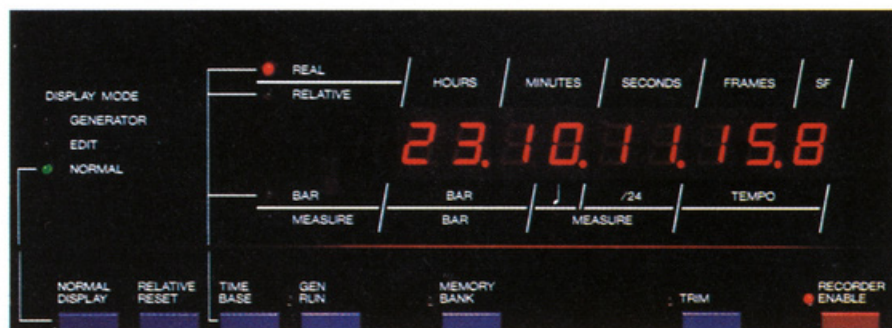
Alcuni sintetizzatori generano 32 note, altri 16, il vostro Amiga, usato come generatore musicale ha 4 voci, quindi una polifonia a 4 note.

A questo punto introduciamo il concetto di timbro.

Ogni generatore di suono mette a disposizione dell'utente numerossissimi timbri strumentali, che vengono utilizzati dalle voci per ottenere le note con i suoni del preset da voi scelto.

Chiarito tutto questo, che a molti di voi potrà sembrare ovvio e banale, possiamo ora suddividere i generatori di suono disponibili in commercio in due grosse categorie:

1- Quelli in cui tutte le voci emettono note con un unico timbro strumentale (che può ovviamente essere scelto in ogni momento fra tutti quelli disponibili), da qualunque canale MIDI arrivino i messaggi per suonare le note.

2- Quelli in cui è possibile suddividere in gruppi (in modo fisso o dinamico) le voci a disposizione ed assegnare ad ogni

tema delle prestazioni ottenibili dai sequencers.

La scelta del tipo di registratore a nastro può essere fatta in funzione di due fattori fondamentali:

Se il generatore di suono (sintetizzatore) non consente generazioni multitimbriche, ovvero le varie parti di un brano registrate su sequencer possono essere eseguite solo una alla volta, allora è indispensabile disporre di un registratore multipista.

Una pista, che sarà incisa per prima, sarà utilizzata per incidervi il segnale di sincronismo che piloterà tutte le registrazioni successive, fungendo da registrazione master per il sequencer.

Sulle altre piste verranno incisi via via i vari suoni relativi alle parti di ciascuna traccia del sequencer.

Se invece disponete di un generatore di suono multitimbrico, essendo questo in grado di dare in uscita BF tutti i timbri delle parti in esso disponibili contemporaneamente pilotato dal sequencer, è sufficiente un comune registratore a nastro, meglio se stereo, per ottenere la registrazione dell'intero brano.

Ovviamente, pur disponendo del segruppo un timbro strumentale diverso ed un canale MIDI diverso in ricezione.

E' evidente che l'ideale per lavorare con il sequencer è il secondo tipo di generatore, anche se è possibile, benchè piuttosto macchinoso, ottenere buoni risultati anche con il primo.

Una ampia parte seguente di questo articolo è dedicata a chiarire, con esempi applicativi, le modalità per eseguire registrazioni con i due tipi di strumenti visti sopra.

Programma sequencer MIDI

Per quanto riguarda la scelta di un programma sequencer MIDI, oggi, essa è forse più vasta di quello che potete aspettarvi e quasi certamente superiore a quella per programmi word processor o database.

Ci sono dei sequencer di tutti i tipi, dai più semplici ed economici a quelli professionali da usare negli studi di registrazione.

Data l'importanza di questa componente, svilupperemo in dettaglio anche il

condo tipo di generatori di suoni, se dovete registrare un brano che, oltre alla "base" ottenuta con strumenti MIDI, richieda suoni di strumenti convenzionali o di una parte di canto, allora non potete proprio fare a meno del registratore multitraccia.

Il sequencer cuore del sistema di registrazione MIDI

Il sequencer si differenzia dal registratore a nastro per i seguenti principali motivi:

- Non viene registrato il segnale audio in uscita dalla tastiera (o da qualunque altro strumento musicale elettronico MIDI) ma il flusso di dati MIDI, che consiste di informazioni digitali (eventi) che rappresentano sia tutto ciò che l'esecutore effettua sullo strumento, che altre eventuali informazioni ausiliarie al processo esecutivo quali clock di sincronismo ecc.

- L'informazione MIDI non viene registrata immediatamente su supporto magnetico ma viene immagazzinata nella memoria RAM del computer, che viene opportunamente gestita dal programma per ricevere e tenere separate le informazioni relative alle varie tracce.

Le informazioni MIDI, completata la registrazione e gli eventuali ritocchi (editing) del brano, potranno poi essere salvate su dischetto sotto forma di file per poterle riutilizzare in futuro.

- Tutti gli eventi della registrazione (tipo di nota, intensità del tocco,tempo, ecc.) possono essere modificati a livello individuale con le funzioni di edit.

I files dei brani musicali generati dai sequencers possono, tramite i programmi di notazione musicale, generare lo spartito relativo sia sul video che sulla stampante.

- Ciascuna traccia che compone la registrazione di un brano, può essere assegnata ad un canale MIDI diverso, che a sua volta, viene associato ad una diversa voce strumentale sul sintetizzatore o expander collegato.

Ciò consente la massima flessibilità perché, mentre il brano è in esecuzione, è possibile cambiare i timbri sullo strumento finchè si è soddisfatti dell'insieme.

- I programmi sequencer dispongono generalmente di molte tracce (da 8 a 48 ed oltre). Ciò rende possibile registrare diverse parti in alternativa e consente rapidamente di ascoltare varie combinazioni fino a trovare la più piacevole.

E' infatti possibile con opportuni comandi accendere o spegnere le varie tracce durante l'ascolto.

- La duplicazione di una traccia è un'operazione immediata e, essendo il contenuto un'informazione digitale, non si perde assolutamente di qualità.

- Possono essere fatti degli intarsi, sostituendo parti di tracce con altre, o tagli od aggiunte a qualunque evento.

- Diventa semplice la realizzazione di un brano in quanto, se ci sono delle parti ripetitive, è sufficiente eseguirle una sola volta e poi concatenarne le copie in forma di SONG variando eventualmente solo alcuni dati essenziali come i comandi di program change che provocano il cambio dei timbri degli strumenti che devono eseguire quella parte.

- Non è necessario eseguire il brano esclusivamente in tempo reale come avveniva quando si registrava su nastro.

Generalmente i sequencers danno anche la possibilità di caricare gli eventi musicali in modo STEP BY STEP ovvero uno alla volta, mediante la tastiera del computer, consentendo così ai musicisti meno esperti, oppure in caso di passaggi musicali particolarmente difficili, di risolvere brillantemente la situazione.

- Alcuni programmi sequencer consentono di visionare il materiale elaborato o sotto forma di rappresentazione grafica o semplicemente come lista di eventi, in modo analogo ad un normale word processor o addirittura in notazione musicale.

In queste condizioni è particolarmente facile eseguire le modifiche dell'elaborato a video.

- Per venire poi incontro ai neofiti dalla mano ancora incerta, i sequencer dispongono generalmente della funzione "Quantizzazione" o "Autocorrect" che, correggendo i piccoli fuori tempo di certe note, le riporta a tempo, dando una quadratura perfetta all'esecuzione.

E se qualcuno volesse invece lasciare quell'effetto umano alle sue esecuzioni? In questo caso basta non inserire la funzione di autocorrect e suonerete come vorrete.

- Il brano registrato può essere traslato in qualunque tonalità

- Con la funzione di sincronismo esterno, comunemente presente sui sequencer, è possibile sincronizzare la velocità di esecuzione o di registrazione dei brani con apparecchiature esterne come ad esempio una Drum Machine o od una traccia di sincronismo proveniente da un registratore a nastro sul quale sono gia presenti o vi andranno incise altre parti musicali.

- Per registrare su nastro magnetico il prodotto finito, disponendo di un sintetiz-

zatore o di un expander multitimbrico, basta fare funzionare il sequencer in play e, collegando le uscite BF dello strumento ad un comune registratore a cassetta o a bobine il gioco è fatto.

Se le vostre esigenze sono più sofisticate, potete usare un registratore multipista e ripartire la registrazione su varie tracce, aggiungendo successivamente il suono di strumenti non MIDI (ne rimangono pochi ormai) e il cantato.

- I progettisti dei sequencers, oltre a tutte le straordinarie possibilità finora esaminate, si sono via via sbizzarriti ad infilare nei loro prodotti un gran numero di funzioni accessorie che, oltre che utili, qualificano il prodotto in un clima ormai di grande concorrenza.

Alcuni esempi sono:

- Eco MIDI, che dà la possibilitàdi ribattere le note programmandone il numero di volte, la loro distanza temporale nonchè l'intensità individuale.

- Metronomo acustico e visivo che può essere attivato durante la registrazione, stabilendo anche se e quante battute deve scandire prima dell'inizio della registrazione.

- Monitoraggio dinamico con l'imitazione delle colonnine LED degli amplificatori HI-FI per indicare il volume relativo di ogni traccia e quindi consentirne piu facilmente la regolazione.

- Possibilità di modificare a piacere i colori delle schermate video.

- Pagina di block notes video richiamabile in qualunque momento per scrivervi appunti sul lavoro in corso. La pagina ovviamente si può salvare nello stesso file dei dati musicali e vi ricorderà in seguito le vostre osservazioni.

- Possibilità di indirizzare il contenuto delle tracce del sequencer ai generatori di suono interni al computer. Questa opzione è interessante perché se per caso non aveste disponibili gli strumenti MIDI potete ugualmente lavorare seguendo la vostra ispirazione anche se limitatamente al numero di voci disponibili sul computer.

- Possibilità di suonare direttamente le note di un expander MIDI premendo opportuni tasti sulla tastiera del computer.

- Possibilità di ottenere il MERGE dei messaggi in ingresso al computer provenienti dalla tastiera e quelli generati dal sequencer, verso un expander o un sintetizzatore, in modo da ascoltare in contemporanea sia la parte che si sta registrando che quelle gia presenti sul sequencer.

Pensiamo sia meglio fermarci qui per ora su questo tema e, se siete riusciti a seguirci fino a qui, vuol dire che il sequencer vi interessa davvero e allora passiamo a qualche considerazione pratica.

La scelta di un sequencer

Diamo qui per scontato che i lettori abbiano gia investito in un computer Amiga o siano in procinto di farlo, quindi escludiamo dal discorso i sequencer dedicati prodotti dai costruttori di strumenti musicali che, in fondo, non sono poi altro che dei computer con un unico ruolo ben definito: fare il sequencer.

Per nostra fortuna invece molte software houses hanno sviluppato (o come nuovo prodotto o come evoluzione di progetti gia esistenti per altri computer) una gamma di programmi che, pur avendo in comune un certo numero, minimo, delle prestazioni essenziali esposte sopra, si differenziano o per maggiore completezza o per altre prestazioni originali che li caratterizzano rispetto ai concorrenti.

La scelta quindi, come per qualunque altro prodotto industriale elettronico, viene orientata da fattori soggettivi che possono essere influenzati sia dalla quantità che dal tipo di prestazioni e dal modo più o meno semplice o intuitivo di implementarle e di rappresentarle, nonchè dal prezzo.

Indubbiamente il sequencer è uno tra i programmi più avvincenti e questo è confermato dall'abbondanza di pacchetti proposti dalle software houses.

Per contro, ogni sequencer ha le proprie modalità operative e quindi, solo dopo un po' di sperimentazione è possibile capirne a fondo e valutarne le prestazioni in modo comparativo,anche se oggi, un pacchetto di prestazioni da considerarsi standard è ormai la costante di tutte le proposte. E' comunque nostra intenzione offrirvi prossimamente una selezione e recensioni di programmi sequencers per aiutarvi nelle vostre scelte.

Esempi applicativi per registrazioni a nastro tramite MIDI

Per concludere la trattazione teorica, pensiamo utile illustrare con qualche esempio i set-up tipici che si possono allestire in funzione degli "attrezzi"di cui disponete.

Il set-up relativo allo schema 1 è da considerare se disponete di una tastiera multitimbrica che consenta di ricevere separatamente ed in contemporanea diverse parti strumentali su altrettanti canali MIDI e di associarli a canali diversi.

Re registrazione delle singole tracce MIDI del sequencer, una per ogni parte strumentale del brano, viene fatta con la tastiera.

I dati MIDI passano dall'uscita MIDI OUT della tastiera e giungono alla porta MIDI IN dell'interfaccia che li invia al computer.

In fase di ascolto, tutti i dati di tutte le tracce del brano vengono inviati, con processo inverso, dal computer alla presa MIDI IN della tastiera.

Questa esegue il brano usando tutte le timbriche che sono state predisposte e quindi possono essere registrate su nastro.

Nello schema 2 èraffigurato il layout che coinvolge una tastiera, un expander multitimbrico ed una DRUM MACHINE.

Rispetto al caso precedente , dovuto al gran numero di uscite BF da gestire, diventa indispensabile l'uso di un mixer.

La tastiera, anche se di tipo economico, (non multitimbrica su diversi canali MIDI), è usata per fare l'input dei dati MIDI relativi all'esecuzione musicale sulle tracce del sequencer .

Per la fase di PLAY, assegneremo a video ogni traccia a un rispettivo numero di canali MIDI, corrispondenti ciascuno ad altrettanti timbri che avremo predisposto sull'expander in ricezione.

Disponendo di sequencers con funzionamento step by step, possiamo anche fare a meno della tastiera, utilizzando così l'expander solo per generare i suoni prodotti dal sequencer e non in tempo reale.

E se disponiamo solo di una tastiera che consente di generare un solo timbro alla volta?

Si può ancora usare con profitto il sequencer MIDI, anche se il processo diventa un pochino più macchinoso.

In questo caso è comunque indispensabile munirsi di due cose fondamentali:

- un registratore a nastro multipista (a

Il Synclavier Digital Audio System ed il Direct to Disk Multitrack Recorder costituiscono il "tapeless studio", un "ambiente" completamente computerizzato per registrazioni musicali professionali a cui non occorre il registratore a nastro

cassetta o bobine)

- un convertitore di segnale di sincronismo (MIDI/FSK e viceversa)

L'esigenza del primo oggetto è ovvia in quanto, se la vostra tastiera produce solo un timbro alla volta, se vogliamo sentire un quartetto dobbiamo incidere sul nastro le quattro parti in tempi successivi.

Si, ma allora perché dobbiamo usare il sequencer MIDI?

E' molto semplice, voi dovete registrare ugualmente i dati MIDI di ciascuna parte su altrettante tracce del sequencer, in modo da poterne utilizzare sia tutti i vantaggi descritti in precedenza che per ottenere la sincronizzazione fra le varie parti del brano.

Quando tutte le parti saranno pronte, allora, inviando il contenuto di ogni traccia MIDI (una alla volta) al vostro sintetizzatore, ne inciderete il suono corrispondente su altrettante piste del nastro.

Se non fosse ancora chiaro, diciamo che se ad esempio non foste soddisfatti del timbro inciso su di una pista, basta fare eseguire di nuovo al sequencer la traccia corrispondente cambiando il timbro (lo stesso vale per eventuali correzioni alle note).

Tutto il lavoro ripetitivo quindi lo fa la macchina.

Resta ancora da esaminare un piccolo ma importante dettaglio per fare funzionare questo tipo di sistema: la sincronizzazione fra le varie parti musicali incise via via sul nastro e quelle del sequencer.

Se ci pensate bene infatti, dopo aver inciso la prima parte su nastro, come fate a garantire che effettuato il riavvolgimento del nastro, al momento dell'incisione della seconda parte riuscirete a fare partire insieme il sequencer ed il registratore?

E' praticamente impossibile, figuratevi poi se avete anche una DRUM MACHINE esterna da tenere a bada!

La soluzione è una sola: affidare la sincronizzazione del sistema al registratore a nastro.

Ciò si ottiene incidendo prima di ogni altra cosa una pista del nastro con il segnale di sincronismo e poi derivando da questa pista (in ascolto) il sincronismo per pilotare sia il sequencer che l'eventuale DRUM MACHINE mentre incidete le altre piste.

E' evidente che una pista del nastro va persa ai fini audio, ma non c'è altra solu-

zione.

Qui entra in campo il secondo oggetto di cui parlavamo: il convertitore di segnale di sincronismo.

Infatti il segnale di sincronismo MIDI, essendo costituito da un segnale digitale, non è direttamente registrabile sui registratori a nastro audio.

Occorre pertanto convertirlo in un segnale audio FSK (Frequency Shift Keying) che sia in grado di essere interpretato dai registratori a nastro, che sono apparati analogici.

Questo compito viene quindi affidato al convertitore che lo svolge sia in un senso che nell'altro da buon interprete.

I collegamenti di questo componente sono illustrati nello schema 3.

Per chiarire meglio le fasi attraverso cui occorre passare per questo processo, la sequenza degli schemi 4, 5a, 5b e 6 ve ne illustra le modalità operative.

Nello schema 4 tramite la tastiera si registrano in sequenza le varie parti strumentali del brano sulle tracce del programma sequencer.

Sulla DRUM MACHINE si imposta una SONG ritmica da abbinare al brano.

Il passo successivo può prevedere due alternative:

- Nella prima, schema 5a, se utilizzate una DRUM MACHINE, mentre si esegue l'ascolto della song di batteria, il relativo sincronismo MIDI (emesso puntualmente dalla macchina),covertito in segnale audio FSK viene registrato sulla prima pista del nastro.

- Nella seconda, schema 5b, se non utilizzate una DRUM MACHINE, potete incidere la pista di sincronismo direttamente dal sequencer mentre ascoltate l'esecuzione di una traccia qualsiasi, su cui avete in precedenza registrato ad esempio una parte di basso tramite la tastiera.

Possiamo ora passare alla registrazione con lo schema 6.

Il sequencer deve essere presettato per ricevere il sincronismo esterno e per leggere una traccia.

Nel registratore mettiamo in ascolto la pista 1 ed in registrazione la pista 2.

Quando il registratore parte, dalla prima pista il sincronismo arriva tramite il convertitore e l'interfaccia MIDI al sequencer che entra in azione, inviando all'uscita MIDI OUT dell'interfaccia i dati verso la tastiera. Questa esegue la parte strumentale (specificata dalla traccia del sequencer) ed il segnale audio corrispondente viene inciso sulla pista 2 del registratore.

Analogamente si ripete l'operazione per le altre tracce del sequencer verso le piste del registratore.

Con il programma Sound Scape pro MIDI della Mimetics Corporation - USA, è possibile, oltre che disporre di uno studio audio MIDI, lavorare anche sul segnale video in arrivo da telecamera o videoregistratore e mixarlo con animazioni grafiche realizzate col computer

CALENDARIO CORSI 1989

AREA INFORMATICA, AUTOMAZIONE D'UFFICIO E INTELLIGENZA ARTIFICIALE

AUTOMAZIONE D'UFFICIO E AMBIENTE P.C.

	CODICE	CORSO	ORE	PREZZO (1)	GENNAIO	FEBBRAIO	MARZO	APRILE	MAGGIO	GIUGNO	LUGLIO	AGOSTO	SETTEMBRE	OTTOBRE	NOVEMBRE	DICEMBRE
N	AU-20	Conoscere l'informatica e il P.C.	40	900.000	23-27								4-8			
	AU-02	Word	24	450.000			1-3								6-8	
	AU-04	Lotus 1-2-3	24	500.000		1-3								2-4		
	AU-05	Symphony	40	700.000			6-10								13-17	
	AU-06	DBIII Plus utenti	24	450.000	30---------	1								25-27		
N	AU-16	Excel	24	700.000						19-21						11-13
N	AU-11	Programmazione windows base	80	2.000.000				10-21								11-22

LINGUAGGI DI PROGRAMMAZIONE

	CODICE	CORSO	ORE	PREZZO (1)	GENNAIO	FEBBRAIO	MARZO	APRILE	MAGGIO	GIUGNO	LUGLIO	AGOSTO	SETTEMBRE	OTTOBRE	NOVEMBRE	DICEMBRE
	PE-02	Basic	40	600.000		20-24								23-27		
	PE-03	Pascal-Turbopascal	48	750.000				3-10								11-18
	PE-05	Cobol	56	850.000			13-22								20-28	
	PE-06	Linguaggio "C"	80	1.400.000	16-27					12-23			11-22			
N	PE-06A	Ottimizzazione e debugging "C"	40	1.300.000			6-10								6-10	
	PE-08	Prolog-Turboprolog	40	1.200.000							10-14		25-29			
	PE-10	Lisp	40	1.050.000					15-19					9-13		

SISTEMI OPERATIVI

	CODICE	CORSO	ORE	PREZZO (1)	GENNAIO	FEBBRAIO	MARZO	APRILE	MAGGIO	GIUGNO	LUGLIO	AGOSTO	SETTEMBRE	OTTOBRE	NOVEMBRE	DICEMBRE
	AU-01	MS-DOS e ambiente P.C.	24	450.000	13-15					26-28			25-27			
N	AU-12	OS/2 - Architettura	40	1.200.000				17-21						23-27		
N	AU-25	OS/2 - Presentation manager	40	1.200.000					8-12							18-22
	I-09	Unix-Xenix utenti	56	1.400.000					22-30						6-14	

DATA BASE

	CODICE	CORSO	ORE	PREZZO (1)	GENNAIO	FEBBRAIO	MARZO	APRILE	MAGGIO	GIUGNO	LUGLIO	AGOSTO	SETTEMBRE	OTTOBRE	NOVEMBRE	DICEMBRE
	AU-07	DBIII Plus programmazione	24	500.000				3-5								4-6
N	AU-08	DBase IV	32	700.000						19-22			4-7			
N	AU-21	Focus utenti	24	700.000					8-10					9-11		
N	AU-22	Focus programmazione	40	900.000							3-7				13-17	
N	AU-23	Oracle	40	1.000.000						5-7						11-15

METODOLOGIE INFORMATICHE

	CODICE	CORSO	ORE	PREZZO (1)	GENNAIO	FEBBRAIO	MARZO	APRILE	MAGGIO	GIUGNO	LUGLIO	AGOSTO	SETTEMBRE	OTTOBRE	NOVEMBRE	DICEMBRE
	PE-01	Programmazione elettr. corso base	96	1.500.000			28------	12					4-19			
	I-102	Case	40	2.000.000					8-12					2-6		
	I-105	Ingegneria del software	80	2.000.000		6-17							11-23			
N	I-111	Software quality assurance	32	1.200.000		13-16								16-19		
	I-108	Architettura SNA	32	1.500.000							5-8				27-30	

INTELLIGENZA ARTIFICIALE

	CODICE	CORSO	ORE	PREZZO (1)	GENNAIO	FEBBRAIO	MARZO	APRILE	MAGGIO	GIUGNO	LUGLIO	AGOSTO	SETTEMBRE	OTTOBRE	NOVEMBRE	DICEMBRE
	IA-01	Intellig. Artif. - corso base	40	1.400.000				17-21							20-24	

DESK TOP PUBLISHING

	CODICE	CORSO	ORE	PREZZO (1)	GENNAIO	FEBBRAIO	MARZO	APRILE	MAGGIO	GIUGNO	LUGLIO	AGOSTO	SETTEMBRE	OTTOBRE	NOVEMBRE	DICEMBRE
	DTP-1	Desk top publishing base	40	1.400.000			6-17								27-------	1
	DTP-2	Ventura	24	600.000						26-28						4-6
	DTP-3	Page maker	24	600.000				12-14								18-20
	DTP-4	Manuscript	24	650.000					29-31					23-25		

COMPUTER GRAFICA E MEMORIE OTTICHE

	CODICE	CORSO	ORE	PREZZO (1)	GENNAIO	FEBBRAIO	MARZO	APRILE	MAGGIO	GIUGNO	LUGLIO	AGOSTO	SETTEMBRE	OTTOBRE	NOVEMBRE	DICEMBRE
N	CG-01	Introduz. ai CD-Rom e videodischi	40	1.300.000						5-9			25-29			
N	CG-02	Concorde	40	1.000.000		6-10								9-14		
N	CG-03	Computer grafica	80	2.000.000				3-14								11-22
	AU-10	Autocad	32	900.000			13-16								13-16	

AREA TELECOMUNICAZIONI - TELEMATICA

	CODICE	CORSO	ORE	PREZZO (1)	GENNAIO	FEBBRAIO	MARZO	APRILE	MAGGIO	GIUGNO	LUGLIO	AGOSTO	SETTEMBRE	OTTOBRE	NOVEMBRE	DICEMBRE
N	T-I1	Il Pabx: Strutture e nuove utilizzazioni	40	1.500.000											6-10	
	T-I2	Tecniche base di trasmissione PCM	40	1.500.000												11-15
	T-I3	Tecniche base e sistemi per trasmissione dati	80	1.700.000		20------	3						18-29			
	T-I4	Apparati e sistemi per le reti di computer	40	1.500.000			13-17							9-13		
	T-I5	Reti a commutazione di pacchetto	40	1.700.000		6-10								16-20		
	T-I6	L'integrazione nelle reti di TLC (ISDN, BX)	32	1.800.000											27-30	
	T-I7	Servizi a valore aggiunto sulle reti X25	24	1.000.000		15-17										4-6
	T-I8	Il modello OSI	32	1.500.000		15-17									20-23	
	T-I9	Architettura SNA: principi e applicazioni	40	1.800.000				10-14								

CODICE	CORSO	ORE	PREZZO (1)	GENNAIO	FEBBRAIO	MARZO	APRILE	MAGGIO	GIUGNO	LUGLIO	AGOSTO	SETTEMBRE	OTTOBRE	NOVEMBRE	DICEMBRE
AREA ELETTRONICA															
EM 1	Criteri di progettazione analogica	40	700.000	16-20			3-14*		12-16				9-20*		
EM 2	Criteri di progettazione digitale	40	700.000	23-27			17-28*		19-23						
EM 3	Progettazione con micro a 8 bit	40	700.000	30-----	3 27---	-10*			26-30			4-19*			
EM 4	Progettazione con micro a 16 bit	40	1.000.000		6-10		3-14*			3-7				6-17*	
N EM 5	Progettazione con micro a 32 bit	40	1.300.000				3-7			3-14*					
N EM 6	Progettazione con micro single chip	40	1.000.000				17-21								
N EM 7	Periferiche per microprocessori	40	1.000.000				10-14								
N EM 8	Uso e applicazioni delle memorie	40	1.000.000							3-14*					
N EM 9	Collaudo ATE	40	700.000					22-26							
N EM 10	Convertitori A/D D/A	40	700.000						5-16*			4-8			
N EM 11	Progettazione e normative di sicurezza	24	600.000										2-4		
N EM 12	Progettare in alta frequenza	40	700.000			6-10								6-10	
N EM 13	Progettazione alimentatori	20	500.000		6-10*					10-14				6-17*	
N EM 14	Strumentazione e controlli industriali	40	1.000.000							17-21					
N EM 15	Uso di PC industriali	40	1.000.000							24-28				23-------	----6*
N EM 16	Video processor	40	1.000.000										2-6		
N EM 17	Progettazione di ciruiti ASIC	40	2.200.000			13-17		8-12	5-16*	10-14		11-15	9-20*	13-17	4-15*
N EM 18	Applicazione delle logiche programmabili	40	1.000.000				8-19*	5-19						13-24*	
N EM 19	I micro e il linguaggio C	40	1.000.000		13-17							11-15			
N EM 20	Processori di segnali digitali	60	2.000.000			14-22						18-27			
N EM 21	Processori a filtri programmabili	40	1.700.000												
N EM 22	Cad Cam	40	1.300.000										16-20		
N EM 23	Reti di microcomputer	40	1.000.000					22-26*							
N EM 24	Progettazione dei BUS	20	700.000			13-17*									
N EM 25	Sistemi operativi per microprocessori	40	1.000.000											20-24*	
N EM 26	Manutenzione Personal Computer	40	1.000.000							3-14*				27-------	----1
AREA AUTOMAZIONE INDUSTRIALE E ROBOTICA															
AI&R 1	Specializzazione in automazione e robotica		2.500.000									4------	----------	---27*	
AI&R 2	Microprocessori a 8 bit	40	700.000									4-19*			
AI&R 3	Traduttori sensori attuatori	20	500.000									21-28*			
AI&R 4	Controllori logici programmabili	40	700.000										2-10*		
AI&R 5	Controllo numerico	20	500.000										19-26*		
AI&R 6	Elementi base di robotica	20	500.000										30----	-----7*	
AI&R 7	Strutture informatiche nel processo produttivo	20	500.000											9-16*	
AI&R 8	Reti di comunicazione nella fabbrica	20	500.000											20-21*	
AREA ALTE TECNOLOGIE SPECIALI															
N ATS 1	Criteri di prova vita sui semiconduttori	40	1.000.000					15-19				11-22*			
N ATS 2	Costi della qualità	8	250.000		22					5					
ATS 3	Tecnologie in fibra ottica per TX dati e immagini	40	1.000.000					8-19*							
ATS 4	Progettazione dei moderni circuiti stampati	40	1.000.000				3-7*								
ATS 5	Affidabilità, circuiti e componenti elettronici	40	1.000.000	13-17*											
ATS 6	Tecnologie VLSI	40	1.300.000*												11-15

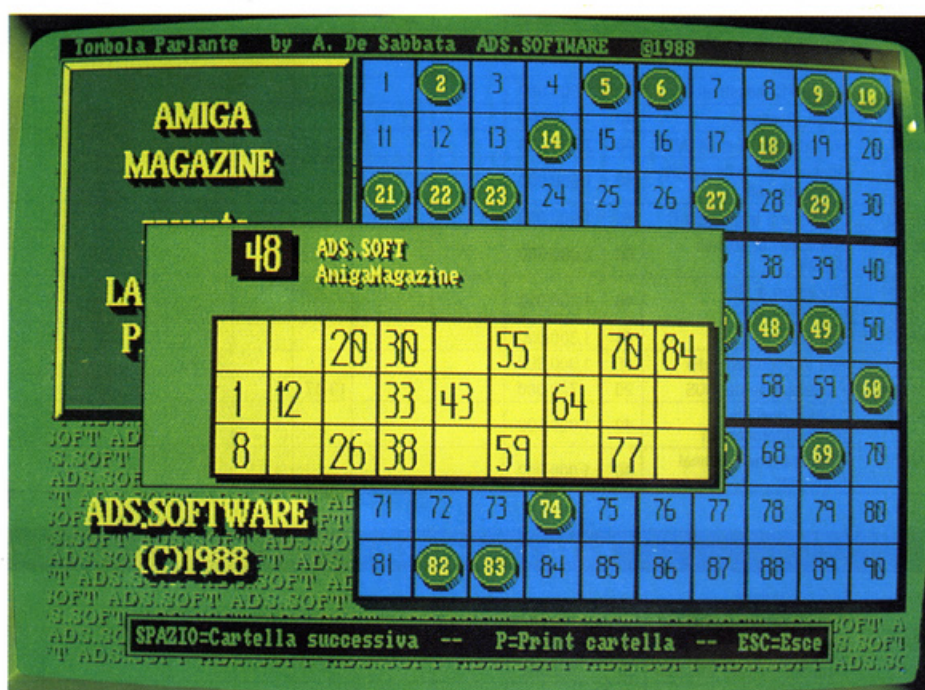
N = Corsi attivati nell'89.
(1) = Iva esclusa - È compresa la fornitura di testi Jackson e dispense Jackson Sata e servizio mensa.
* = Sono previste sessioni serali in data da definire con gli interessati.

SCUOLA DI ALTE TECNOLOGIE APPLICATE

S.A.T.A.

Per le modalità di iscrizione e richiesta programmi dettagliati scrivere o telefonare alla DIVISIONE FORMAZIONE E PRODOTTI PER LA DIDATTICA del Gruppo Editoriale Jackson

TOMBOLA

Anche col basic si può...

di A. De Sabbata

Dimostreremo a chi avrà la pazienza di seguirci, che la programmazione non presenta, anche per i principianti, difficoltà insormontabili, e specialmente con il nostro Amiga, dotato di un basic estremamente efficace, all'atto pratico, tutto si rivela molto più facile del previsto. Per chi è alle prime armi infatti, non c'è niente di meglio che sforzarsi di scrivere qualche breve programma per acquistare una certa padronanza nel linguaggio usato.

1) Creare una schermata grafica con il nostro "Paint" preferito, ed utilizzarla nei nostri programmi basic.
2) Mescolare un sacchetto di numeri oppure un mazzo di carte.
3) Scrivere numeri giganti sullo schermo.
4) Migliorare la pronuncia del nostro Amiga: di fatto costruiremo un traduttore che sostituirà la funzione di sistema TRANSLATE per le frasi in Italiano.
5) Usare alcune preziose funzioni disponibili nelle librerie.
6) Stampare la window corrente, (con o senza i bordi).

Questi sono alcuni obiettivi che potremo raggiungere assieme armeggiando col (potente) basic del nostro Amiga. Per ottenere il nostro scopo realizzeremo alcune routine, che non rimarranno fini a se stesse, ma potranno essere utilizzate da chiunque in propri programmi. Nel nostro caso, faranno tutte parte di un programma, poco più che banale, (il gioco della Tombola), ma che serve egregiamente allo scopo dell'articolo.

Crediamo che se per i lettori di una certa età questo gioco potrà far rivivere remoti ricordi di quiete serate di una oramai passata gioventù, altresì per alcuni fra i piu giovani dei seguaci di Amiga Magazine, sarà una (speriamo piacevole) scoperta. Abbiamo modo di credere infatti, che la Tombola, tanto amata qualche decennio orsono, sia pressochè sconosciuta tra i giovani d'oggi.

Rimbocchiamoci quindi le maniche, e diamo inizio alla nostra carellata, e non scandalizzatevi se ci aiuteremo scopiazzando di brutto tra i file della directory "Ba-

sicDemos" contenuta sul dischetto EXTRAS in dotazione ad ognuno, (sono lì per quello!).

Questa directory, si rivela infatti una miniera di ottimo materiale, il quale può essere usato così com'è, o in certi casi adattato a scopi particolari.

I programmi di cui ci serviremo direttamente nel corso del nostro lavoro, e che entreranno a far parte (più o meno modificati), del codice sorgente del programma che costruiremo sono:

LoadACBM Serve a caricare una schermata precedentemente disposta per i nostri scopi.

ScreenDump Serve a produrre su stampante la schermata attualmente visualizzata. Noi vedremo come con pochissime modifiche, lo stesso programma potrà essere utilizzato per effettuare il DUMP della sola finestra attualmente in uso.

Altri programmi forniti col dischetto EXTRAS, sono stati utilizzati per raggiungere alcuni scopi particolari, che in seguito esamineremo.

Come creare la schermata su cui lavorare

Per la maggior parte dei programmi, anche se scritti in BASIC, è preferibile preparare la "maschera" che costituirà la schermata di lavoro, con un programma dedicato al disegno, invece che tramite le varie istruzioni grafiche dello stesso basic. Questo infatti permetterà al nostro programma di presentarsi con una veste particolarmente gradevole, e rifinita anche nei più piccoli dettagli.

Ci metteremo quindi all'opera con il programma che più soddisfa le nostre esigenze, e disegneremo la pagina di lavoro. In seguito, terminato il disegno, lo dovremo salvare, e questo sarà di norma trasformato in un file dal formato standard IFF (Interchange File Format).

Per nostra sfortuna, però questo formato che offre indiscutibili vantaggi, quali la piena compatibilità con qualsiasi programma che usi questo standard, nonchè la compattezza del file contenente i dati, è alquanto sconsigliabile per l'uso in programmi basic a causa della esasperante lentezza con cui questo viene caricato in memoria. Per convincersi di questo è sufficente, provare ad usare il programma "LoadILBM-SaveACBM" contenuto sempre nella directory BasicDemos di EXTRAS.

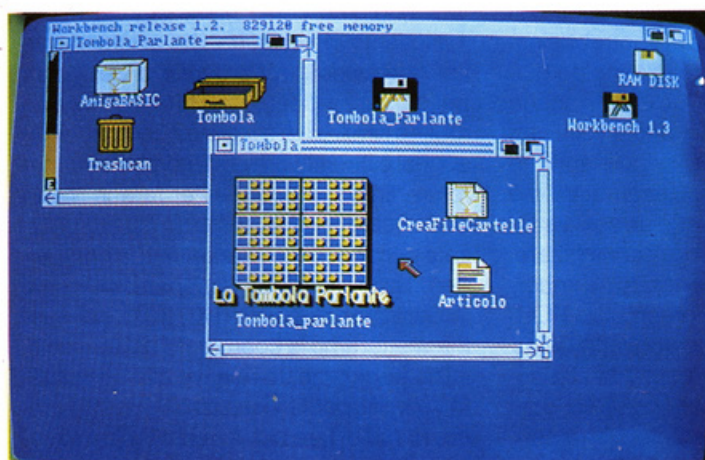
Con questo programma, è possibile oltre che caricare e visualizzare un file grafico in formato IFF (ILBM), anche salvare lo stesso file nel formato ACBM, (Amiga Contiguos BitMap), più idoneo ad essere utilizzato dal nostro Basic, anche se gli stessi dati occuperanno ora un maggior spazio sul disco, non essendo sottoposti ad alcun compattamento. Ottenuto il file dal formato ACBM, si potrà notare la velocità con cui ora viene caricata in memoria e visualizzata la schermata. Per fare questa prova si usi il programma LoadACBM.

A questo punto, è doveroso aprire una parentesi, per chiarire alcuni particolari che in certi casi potrebbero creare qualche difficoltà.

I programmi di cui si è parlato più sopra, e cioè LoadACBM e LoadILBM-SaveACBM, sono strutturati in modo da aprire un nuovo schermo dalle dimensioni uguali alla figura da visualizzare, e su di questo una finestra a tutto schermo. Le informazioni di cui hanno bisogno, sono infatti contenute nei dati della figura stessa. Può capitare però il caso di voler aprire la propria finestra di lavoro sullo schermo già esistente del WorkBench. Questo è necessario, se si vuole risparmiare preziosa memoria RAM, quando il programma ne avrà bisogno per inizializzare altre strutture, usare più fonti carattere eccetera.

Un problema con l'AMIGAbasic

Questo è esattamente quanto è capitato durante la stesura del programma "Tombola Parlante". Infatti, lavorando disinvoltamente su di un 2000, (e senza esperienza del mondo AMIGA), si è portato a termine senza problemi un programma, che pur essendo di dimensioni estremamente esigue, necessitava per girare di più di 500k di memoria! Bisogna infatti tener presente degli oltre 100k che servono a mantenere in memoria l'interprete, della memoria occupata dallo schermo e dalla finestra (creati in alta risoluzione, 600 x 256, con otto colori ed il refresch), dalle librerie aperte, nonchè da due ulteriori fonti carattere di cui si è usufruito!

Era chiaro che volendo pubblicare il programma era necessario adattarlo alla memoria disponibile sull'Amiga 500 inespanso. Perciò si sono cercate varie soluzioni, e alla fine si è giunti ad un secondo programma, che per occupare meno memoria usufruisce dello schermo del WorkBench, su di una finestra di dimensioni

minori! Le differenze tra i due riguardano infatti esclusivamente la presentazione grafica. Il primo offre una schermata piena, (640 x 256) mentre il secondo si deve accontentare di 640 x 200 punti. È noto infatti che a causa di un bug l'AMIGAbasic pur comportandosi normalmente su schermi creati dall'utente, si rifiuta di aprire finestre che vadano al di sotto dei fatidici 200 punti ammessi dallo standard Statunitense, vedi infatti la propria finestra di output.

Sembra che neanche le funzioni di Intuition WindowLimits e SizeWindow abbiano alcun potere su di una finestra aperta da basic sullo schermo del WorkBench. Probabilmente, aprendo la finestra tramite la funzione OpenWindow sempre appartenente alla Intuition Library sarebbe possibile aggirare il problema, ma facilmente in quel caso i normali comandi grafici del basic non sarebbero più disponibili nella finestra così ottenuta, a causa di una differente gestione delle Window da parte del basic.

È evidente che le finestre aperte tramite il basic sullo schermo del WorkBench nascono con il parametro "MaxHeight" (altezza massima) impostato a 200. Provate ad impartire in modo diretto nella finestra di output del basic la seguente istruzione, non prima di esservi accertati di aver salvato i programmi in memoria:

```
POKEW WINDOW(7)+22,256
```

A questo punto, puntate il mouse sul gadget di ridimensionamento (in basso a sinistra) della finestra di output del basic, e, sorpresa, la finestra ora obbedisce ai vostri spostamenti, non bloccandosi più ai famigerati 200 punti! Potrete anche spostare il cursore in quello spazio guadagnato, ed immetterci direttamente del testo. Non azzardatevi però a far scrollare lo schermo, (portando il cursore in fondo alla finestra, o impartendo ad esempio il comando FILES), pena un inappellabile GURU. Da programma comunque, sarà sempre impossibile usufruire di quello spazio così poco elegantemente recuperato, in quanto l'interprete controllerà, ad ogni istruzione impartita, che le coordinate richieste non abbiano a superare i 200 punti permessi.

Detto questo, sorge un'ulteriore difficoltà: una volta creato il disegno che ci servirà da maschera di lavoro per il programma in costruzione e salvato su disco, se questo è stato creato con un paint che sfrutta la peculiarità del sistema PAL, (256 o 512 linee), le dimensioni memorizzate assieme ai dati del file grafico (che saranno lette da LoadILBM o LoadACBM), rispecchieranno comunque le dimensioni massime dello schermo usato, anche se il disegno ne occupa soltanto una parte. Si potrebbe pensare di risolvere il problema creando, (tramite lo stesso "paint"), un pennello delle dimensioni volute, ma anche in questo caso, le dimensioni che i programmi LoadACBM e LoadILBM ricaveranno dal file contenente il BRUSH saranno ancora quelle dello schermo su cui il pennello stesso è stato creato. L'unico programma che, (dopo una serie di sperimentazioni, si è rivelato capace di risolvere l'inghippo è il potente THE BATCHER. Questa utility contiene infatti tra le sue opzioni, la possibilità di definire una parte della pagina come una CLIP, sulla quale effettuare tra le altre operazioni anche il SAVE.

Ebbene, la porzione di disegno così ottenuta, è proprio quello che meglio si adatta alle nostre esigenze, e così siamo finalmente riusciti ad ottenere un file in grado di essere letto da LoadACBM, il cui codice, eccetto insignificanti modifiche, è stato riportato tale e quale nella parte iniziale di TombolaParlante.

Diamo una mescolata al sacchetto

Una breve routine che a volte si rende necessaria in molti programmi, è quella di dover creare una struttura di numeri random contenuti in un certo range, ma che non si devono ripetere. Nel nostro caso si tratta del classico sacchetto contenente i numeri da uno a novanta, ma potrebbe essere benissimo un mazzo di carte da gioco o altro. La soluzione al problema anche se estremamente banale, può a volte sfociare, (specialmente nei principianti), in complesse elaborazioni e scambi tra più vettori! È necessario invece un solo vettore, riempito con i numeri da randomizzare, (che siano in ordine o meno non ha importanza), e su di questo con un ciclo FOR-NEXT si opererà degli scambi con l'istruzione basic SWAP. Un'occhiata alla subroutine InitSacchetto utilizzata ad ogni partenza di TombolaParlante, varrà senz'altro più di tante parole.

Scrivere numeri giganti

In alcuni programmi, ed in quello di cui ci occupiamo in particolare, nasce l'esigenza di richiamare l'attenzione dell'utente su particolari caratteri. Nel nostro caso questi sono i numeri estratti, e la routine che ci accingiamo a descrivere è in grado di visualizzare esclusivamente caratteri numerici, ma potrebbe essere facilmente adattata con leggere modifiche a qualsiasi carattere alfabetico.

Probabilmente il modo più efficace per far si che l'ultimo numero estratto sia ben evidenziato sullo schermo è quello di presentarlo notevolmente ingrandito. Esistono di certo un'infinità di modi diversi per ottenere questo scopo, ma dovendoci occupare di un programma in cui la semplicità costruttiva è un fattore determinante, essendo lo stesso rivolto ad attirare l'attenzione delle "matricole" della programmazione, il nostro studio si è rivolto sul particolare modo in cui vengono rappresentate le cifre digitali che ogni giorno abbiamo sotto gli occhi in una miriade di occasioni, dalle calcolatrici tascabili agli strumenti di misura elettronici ecc.

Questi caratteri, sono formati da sette segmenti che si illuminano tutti o in parte in relazione alla cifra da visualizzare. Se noi numeriamo da zero a sei questi elementi, ed inizializziamo la matrice bidimensionale CIFRE(X,Y) con X=9 ed Y=6, in cui ad ogni elemento "X" corrisponderà una cifra, mentre ognuno dei sette elementi "Y" conterrà il valore uno solo se il corrispondente elemento esiste realmente nella cifra in questione, avremo pressochè già risolto gran parte del problema.

Prendiamo come riferimento la figura 1. Notiamo innanzitutto che esistono due tipi di elementi, (orizzontale e verticale), per cui sarà più semplice predisporre due routine simili, che avranno il compito di disegnare il relativo elemento che chiameremo "lineO "e l"ineV". Per fare in modo che queste routine possano essere utilizzate per ognuno degli elementi, a prescindere dalla posizione occupata, dovremo inizializzare, due vettori X() e Y(), nei quali immetteremo le coordinate del vertice di inizio tracciamento relative alle coordinate stabilite per la visualizzazione della cifra. Prima di fare ciò, avremo inoltre asse-

gnato un valore adeguato ad alcune variabili che definiscono il modulo e la grandezza della cifra.

Una particolarità di rilievo che si nota nella figura in esame, è il fatto che la costruzione delle cifre viene costruita su una maglia regolare, di modulo ridefinibile (mx = modulo orizzontale, my = modulo verticale). Anche la lunghezza degli elementi (lx e ly) si basa su un multliplo del modulo. Tutto questo, molto più difficile a dirsi che a farsi, si ottiene con le poche righe alla subroutine InitNumeriGiganti.

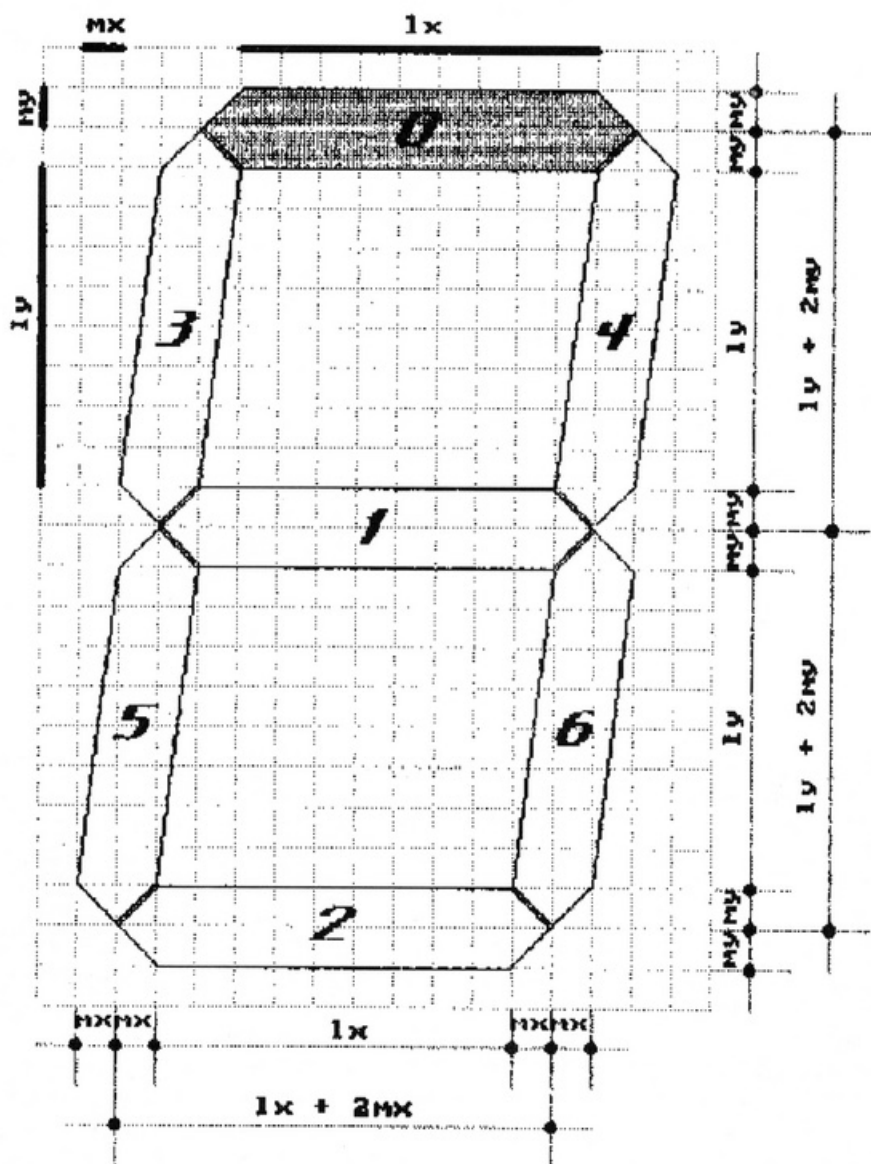
La visualizzazione degli elementi esistenti, (e lo spegnimento di quelli inesistenti se accesi), per ogni cifra, viene effettuato dalla routine NumeriGiganti, e dalle ausiliarie lineO e lineV,come già accennato, tramite le comode istruzione AREA e AREAFILL. Unico inconveniente di queste istruzioni, il fatto di segnalare errore anche se una sola delle coordinate di tutte le istruzioni AREA dovesse superare i limiti della finestra corrente.

Un'ulteriore esempio d'uso della routine NumeriGiganti, è il programmino MegaClock, contenuto nel disco allegato. Questo programma, (che a dire il vero non serve a gran che), simula un'enorme orologio digitale a tutto schermo, prelevando l'ora da TIME$, e visualizzandola nel classico formato HH:MM:SS. Una curiosità che non siamo riusciti a spiegarci: MegaClock funziona egregiamente se interpretato. Provando a compilare il programma con A/C BasicCompiler, non si ottiene altro che un notevole rallentamento, tale da far perdere la visualizzazione di un secondo ogni tre o quattro!

Il pessimo Italiano del nostro AMIGA

Volendo sfruttare il sintetizzatore vocale incluso nella macchina, per ascoltare anche solo brevi frasi in italiano, molti di noi saranno rimasti delusi dall'inflessione tipicamente anglosassone con cui vengono pronunciati i messaggi. La funzione TRANSLATE$ infatti, traduce la frase ricevuta in fonemi adeguati alla corretta pronuncia dell'inglese, incurante del fatto che noi usiamo l'Italiano!

Unico modo per risolvere il problema, o per lo meno per rendere comprensibile quanto declamato dal computer, è quello di procedere alla traduzione in FONEMI delle parole, in base al suono prodotto dalle stesse, bipassando la funzione TRANSLATE$. Fare questa operazione manualmente, anche per brevi messaggi diventa un'operazione noiosissima, oltre che non molto coerente se effettuata da un programmatore o aspirante tale. Non è vero che l'uso più appropriato di un'elaboratore è quello di fargli eseguire compiti ripetitivi?

A questo punto, costretti dal fatto stesso di possedere un computer con sintesi vocale, (che non sa pronunciare la nostra lingua), accingiamoci a scrivere un breve programma che trasformi l'Italiano scritto in fonemi correttamente interpretabili dal comando SAY di AMIGA. La prima cosa da fare è quella di individuare quelle lettere dell'alfabeto che possono crearci problemi. Fortunatamente, questo si risolve in poca cosa: le vocali, comprese quelle accentate, alle quali aggiungeremo "un'indicatore di stress", e la lettera "Z". Per il nostro uso, comprenderemo nell'elenco, anche le cifre da zero a nove, per le quali prepareremo già la traduzione in fonemi.

Esistono però, alcune lettere il cui suono risulta essere diverso, in base al contesto in cui si trovano. Queste sono Q, C, S e G. Per queste lettere, dovremo di volta in volta decidere il fonema corretto, in base ai caratteri che le precedono o le seguono. La lettera "H" invece, la considereremo solo per la sua influenza sul suono delle lettere che la precedono, in quanto essa stessa è sempre muta.

Di tutto questo, si occuperanno le due sezioni "InitRecita" chiamata durante l'inizializzazione, e la subruotine "Recita", la quale analizza a tre a tre i caratteri contenuti in FRASE$, e dopo aver deciso il fonema corretto per ognuno, lo deposita in FON$, che verrà utilizzata con SAY per farci ascoltare la frase così elaborata. Purtroppo, il risultato non è dei più eccellenti, in quanto la cadenza risulta essere senza inflessioni. Un lieve miglioramento si ottiene inserendo opportunamente in alcune parole una vocale accentata, anche se non contenuta dalla parola stessa.

Tradurre un numero nella stringa equivalente

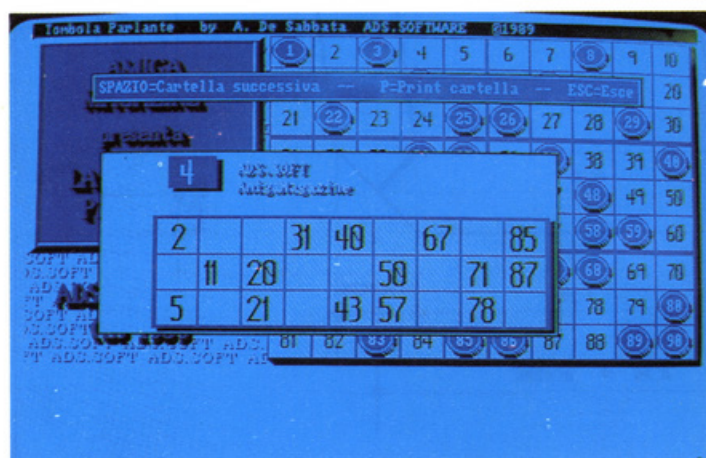
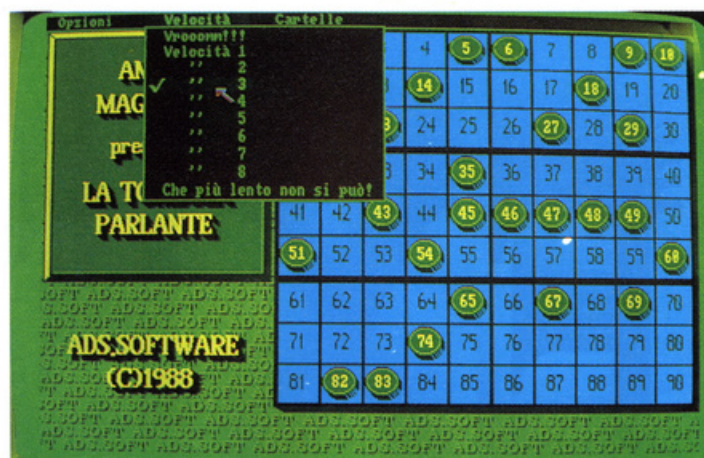
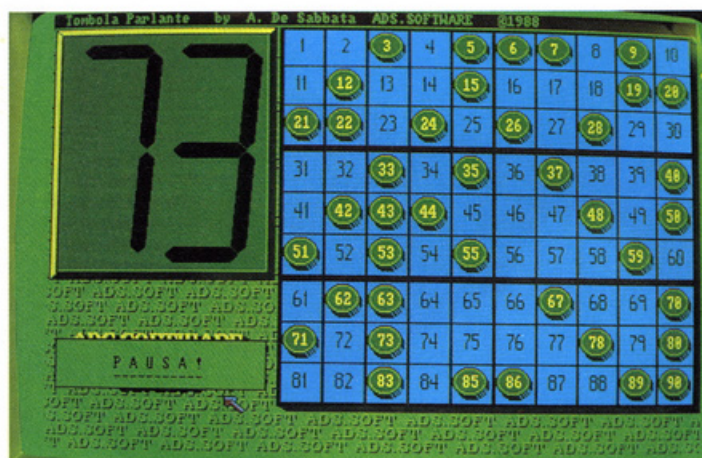
Una difficoltà che sorge volendo ascoltare i numeri estratti, è quella di dover tradurre le cifre numeriche nelle stringhe relative da inviare con "FRASE$" alla subroutine "Recita". Anche per questo lavoro è stato creato un modulo utilizzabile al di fuori di TombolaParlante. Si tratta delle due sezioni "InitTraduttoreNumeri" e "Traduce". Queste ultime sono strutturate per l'uso con numeri da uno a novanta, ma lo stesso algoritmo può benissimo essere utilizzato con poche aggiunte per qualsiasi serie di cifre. Osservando la InitTraduttoreNumeri, potrete in particolare osservare che alcune stringhe contengono qualche erre di troppo. Questo non è un errore, ma un piccolo trucco per alleviare un poco la fastidiosa erre moscia di Amiga.

Le cartelle della tombola

Naturalmente, dovendo giocare alla Tombola, è necessario avere a disposizione un adeguato numero di cartelle. Crediamo che nella gran parte delle nostre famiglie esista una più o meno raffinata versione della tombola, magari dimenticata in qualche polverosa soffitta. Chi non riuscisse proprio a trovarla, potrà creare il numero di cartelle desiderato con questo stesso programma.

È così che arriviamo infine ad una parte alquanto interessante del programma, (naturalmente per il fatto di poter essere riutilizzata), nata dalla volontà di fornire TombolaParlante della possibilità stampare "in casa" le proprie cartelle.

Com'è costruita una cartella

Se qualcuno dei nostri lettori si sarà

soffermato ad osservare con una certa attenzione alcune cartelle della tombola, avrà certo notato il modo bizzarro in cui sono costruite. Ogni cartella contiene 15 numeri che vanno dall'uno al novanta, casualmente disposti su tre righe di nove colonne ognuna. Ogni riga pertanto, dovrà contenere cinque numeri, e necessariamente quattro spazi non occupati. Da osservare però che la casualità della disposizione e della ricorrenza dei quindici numeri nelle ventisette caselle a disposizione è disciplinata da alcune regole. I numeri sono disposti nel seguente modo: dall'uno al nove sulla prima colonna, dal dieci al diciannove sulla seconda, e così via fino all'ultima che conterrà i numeri dall'ottanta al novanta. I numeri che risultano incolonnati nella stessa cartella sono in ordine crescente dall'alto verso il basso e ogni riga contiene non più di due caselle libere adiacenti, lo stesso dicasi per le colonne. Infine, leggendo le cartelle con ordine sequenziale, scopriremo che ogni numero, non si ripete se non dopo un numero minimo di righe. Accertate tutte queste regole, anche se di per se abbastanza semplici, non resta che trasformarle nell'algoritmo di un breve programmino che si occuperà di creare un file su disco nel quale saranno riportati i numeri di ogni riga compressi in una stringa di cinque caratteri per non occupare inutile spazio.

Tutto questo viene eseguito dal programma CreaFileCartelle. Il risultato della elaborazione di questo programma è il file Cartelle.asc che viene letto da TombolaParlante per visualizzare e stampare le cartelle.

Veniamo ora alla routine di stampa. Alla richiesta effettuata tramite il menu, TombolaParlante, aprirà una finestra sulla quale verrà disegnata la cartella numero uno. Con il tasto SPACE potremo visualizzare la cartella successiva, la pressione del tasto "P" provocherà il DUMP della finestra corrente sulla stampante, e la pressione di ESCAPE si provocherà il ritorno al programma principale. Come già acennavamo nell'introduzione dell'articolo, la nostra routine WindowDump non è altro che l'utility ScreenPrint contenuta nella directory BasicDemos di Extras, con solo pochissime modifiche.

Le modifiche apportate riguardano le variabili srcX e srcY le quali originariamente impostate a zero (coordinata in alto a sinistra dello schermo), vengono ora settate alla posizione occupata dall'angolo in alto a sinistra della finestra corrente. MaxWidth e maxHeight che determinano la larghezza e l'altezza massima vengono ora ricavate tramite la funzione WINDOW(). Inoltre, per adattare la routine alle diverse situazioni che possono determinarsi in funzione dei diversi gadget attribuiti alla finestra corrente, sono state create tre nuove variabili denominate wGadget, gadMove e gadSize, grazie alle quali è possibile calcolare il corretto numero di pixel in relazione al valore attribuito alla variabile BorderFlag. Sarà così possibile ottenere il dump della finestra corrente con i bordi o senza, indipendentemente dalle dimensioni dei bordi, le quali dipendono dal valore del parametro "TIPO" (0-31), da riportare nella variabile wGadget della routine WindowDump, utilizzato per creare la finestra. Per l'invio alla stampante saranno utilizzati i parametri settati in PREFERENCES, così che la dimensione della stampa, dipenderà dal numero di caratteri definiti in ricavabili dalla differenza tra Right Margin e Left Margin.

Si sarà notato che il programma TombolaParlante tra l'altro fa uso di alcune delle funzioni della libreria GRAPHICS così da poter avere a disposizione diverse fonti carattere e per disporre il testo in qualsiasi posizione sullo schermo, al fine di ottenere un display più gradevole. Non ci soffermiamo su questo punto, rimandando i lettori all'esauriente articolo "BASIC TEXT" apparso sul numero quattro di Amiga Magazine.

Conclusioni

Per concludere vorremmo far osservare che il programma presentato è volutamente incompleto. Avremmo infatti potuto arricchirlo di una miriade di nuove funzioni, ad esempio il controllo delle uscite, sia sul tabellone che su alcune cartelle selezionate, valutando l'ambo, la terna, la quaterna, la cinquina e la tombola. Si sarebbe potuto abbinare ad ogni numero la definizione data a questo dalla Cabala, oppure una breve frase definibile da ognuno. Noi invece abbiamo voluto fermarci a questo punto per dimostrare come con poco sforzo si riesca ad ottenere anche ottimi risultati, senza dover possedere necessariamente una grande esperienza di programmazione. Del resto, invitiamo i lettori che sono stati attratti dagli argomenti dell'articolo, a completare ed arricchire essi stessi il programma, ribadendo che questo è senz'altro il modo migliore per imparare il basic ed iniziare a programmare.

di Piero Dell'Oste & Marco Arpini

L'AMIGA ha avuto il successo attribuitogli grazie alle sue straordinarie capacità grafiche e sonore. In questa ottica si è un poco trascurata la caratteristica fondamentale che differenzia questa macchina da altre della stessa fascia prezzo-prestazioni. Stiamo parlando del sistema operativo multitasking, o meglio di quel gruppo di funzioni di sistema che permettono all'utente di sviluppare delle applicazioni in multitasking.

Il riferimento principale per tutto quello che diremo in questo articolo è il manuale ROM KERNEL:EXEC. Questo articolo vuole essere un approccio per coloro che non posseggono il manuale in questione e un completamento per coloro che lo posseggono, e che si saranno resi conto della pochezza e incompletezza degli esempi. I nostri programmi sono frutto di esperimenti e quindi non sono necessariamente la via migliore per risolvere i problemi incontrati, saremo ben lieti di ricevere qualsiasi suggerimento che permetta di sfruttare al meglio questa eccezzionale prestazione.

L'architettura del sistema

L'architettura del sistema di Amiga è composta da una struttura simile ad un albero, alla cui testa sono posizionati come pari livello il CLI di AmigaDos e il Workbench con il suo sistema di icone. Il CLI è direttamente collegato con il File System per la gestione dei processi e con il Console Device per la comunicazione dalla tastiera e verso il video. Il File System riceve le direttive dal TrackDisk Device per

AMIGA

tutto ciò che ha a che fare con il drive, e da Exec per quello che riguarda tasks,interrupts,messaggi,I/O(argomento principale dell'articolo). Dall'altra parte il Workbench prende le informazioni necessarie direttamente dal sistema di Intuition, che a sua volta gestisce il tutto tramite i layers e le routine grafiche di basso livello. Al livello più basso di tutto il sistema appena sopra all'hardware puro abbiamo il processore 68000, il controllore del drive, il gestore del mouse e della tastiera, il processore grafico, quello sonoro e per finire le porte di I/O. Ora entriamo nel dettaglio del discorso cominciando a parlare di task.

Task

Praticamente questo è il punto iniziale del discorso; possiamo considerare un task come un insieme di istruzioni che lavorano ad un basso livello nell'architettura del sistema di EXEC. Infatti un task può comunicare direttamente con il processore, ma non può utilizzare funzioni relative al DOS in quanto poste ad un livello superiore. Tali funzioni sono utilizzabili dai processi. Exec riconosce ad un task 3 possibili stati fondamentali:

lo stato di waiting,quello di ready ed infine lo stato di running. Quando un task si trova nello stato di ready è pronto ad utizzare la CPU e sarà Exec a stabilire il momento di accesso, mandando automaticamente il task stesso in stato di running.

Da questo si deduce che soltanto un task tra i tanti in stato di ready può andare in esecuzione. Invece un task in stato di waiting si trova in stallo, di conseguenza non può essere pronto per l'utilizzo della CPU e rimarrà in tale situazione fino a quando un evento esterno lo "risveglierà". L'evento esterno può arrivare da qualunque parte del sistema,ad esempio un task che invia un messaggio ad un altro task,un segnale inviato da un qualunque device,la pressione di un tasto ecc...Exec provvede a mantenere due liste di task, la lista dei task in stato di ready e quella dei task in stato di waiting.

Mentre per lo stato di waiting la lista non ha un particolare ordinamento, la lista di ready viene ordinata in base alla priorità del task stesso, perciò tasks aventi alte priorità si troveranno nelle prime posizioni della lista. La priorità è un parametro molto importante perché permette ad Exec di stabilire la frequenza di accesso di un task alla CPU. Il programmatore ha la possibilità di modificare il valore privilegiando uno o più task rispetto ad altri. Come ottenere questo effetto verrà spiegato più avanti, per ora basta sapere che dovrebbe assumere un valore compreso tra +20 e -20,lo 0 indica la priorità base. Fondamentalmente Exec utilizza strutture di dati, ad esempio la struct Task che verrà esaminata in seguito per compiere le sue operazioni, ed il suo data base per la ricerca di queste strutture si basa su una lista di nodi. Un nodo viene definito dalla seguente struttura:

```
struct Node
  {
   struct Node *ln_Succ;   puntatore al prossimo
          nodo
   struct Node *ln_Pred;   "      " precedente
   UBYTE ln_Type;   tipo del nodo
   BYTE  ln_Pri;                priorit à del nodo
   char * ln_Name;   nome del nodo
  };
```

Il tipo del nodo viene definito con alcuni #define che si trovano nel fileinclude "exec/nodes.h". Un alto valore del campo ln_Pri metter. à il nodo nei primi posti della lista ordinata di nodi. Infine ln_Name è un puntatore al nome del nodo; questo facilita la ricerca dei nodi. La lista dei nodi viene definita dalla struttura riportata sotto:

```
struct List
  {
   struct Node *lh_Head;           puntatore al pri-
                                   mo nodo della lista
   struct Node *lh_Tail;           sempre a 0
   struct Node *lh_TailPred; puntatore all'ulti
                                   mo nodo della lista
   UBYTE lh_Type;    tipo di nodo
   UBYTE lh_Pad;                   non utilizzato
  };
```

Ad ogni task viene assegnata una struct task:

```
extern struct task
  {
   ...
   ...
   ...
   ...
   APTR tc_SPReg;
   APTR tc_SPLower;
   APTR tc_SPUpper;
   ...
   ...
  }
```

Gli altri membri della struttura non sono stati menzionati in quanto la loro spiegazone esula ed appesantisce la trattazione di questo articolo.I campi SPLower,SPUp-

per, SPReg indicano rispettivamente il limite inferiore dello stack, il limite superiore e il punto d'inizio dello stack stesso, che conseguemtemente assume il valore di SPLower. Di tutto questo il programmatore si deve interessare quando intende aggiungere un task al sistema utilizzandola funzione di sistema AddTask(). Questa funzione richiede 3 parametri che nell'ordine sono:
1) un puntatore ad una struct task
2) il nome di una funzione che rappresenta il task da aggiungere
3) il nome di una funzione che verrà chiamata quando il task aggiunto dovrà essere rimosso dal sistema.

Quest' ultimo punto è opzionale, il programmatore può indicare nell' AddTask questo termine pari a 0, in tal caso Exec utilizzerà la sua routine per la rimozione del task ovvero la funzione RemTask(). Chiaramente prima di chiamare la funzione AddTask() si deve allocare una struct Task in modo da ottenere un puntatore a quest' ultima, allocare una zona di memoria per lo stackpointer e conseguentemente riempire i membri SPReg,SPLower,SPUpper che abbiamo precedentemente analizzato. Poi si devono riempire i campi della struct Node allocata nella struct Task e solo ora è possibile eseguire la chiamata alla funzione AddTask().

Il programma "create.c" non compie queste operazioni in quanto utilizza una funzione di amiga.lib CreateTask(name,pri,initialPC,finalPC),che esegue queste operazioni per noi, ritornando se tutto si svolge correttamente un puntatore alla struct task del task aggiunto al sistema. I parametri della funzione Createtask() dovrebbero risultare chiari, se qualcuno avesse ancora dei dubbi gli rammentiamo che tale funzione verrà presentata al termine dell'articolo.

Qualche dubbio può sorgere riguardo alla dimensione dello stack visto che Exec non si preoccupa di gestirlo.Il ROM KERNEL:EXEC consiglia la dimensione di 1K come sufficiente per la gestione di più chiamate a funzioni e il salvataggio dei 17 registri della CPU. L'esperienza ha dimostrato però che tale valore non è sempre sufficiente,infatti nei nostri task aggiunti lo stack è stato settato a 4K perché il valore di 1K provocava il fatidico GURU MEDITATION, e valori intermedi non assicuravano il corretto funzionamento sempre.

Finalmente il task funziona, ma durante questa fase potrebbe richiedere l'utilizzo di risorse che task concorrenti vorrebbero accedere, tutto ciò porterebbe al caos, ma Exec ci fornisce un paio di possibilità per risolvere questo problema. La prima consiste nel bloccare il multitasking utilizzando la coppia di funzioni Forbid() e Permit(). Infatti se un task durante la sua esecuzione chiama la funzione Forbid() può ottenere una qualunque risorsa sicuro che nessun altro task lo disturberà,tutto ciò fino al Permit(). Comunque durante il Forbid()-Permit() un interrupt potrebbe causare un pasticcio, perciò Exec fornisce la coppia di funzioni Disable()-Enable() con le quali blocchiamo gli interrupt oltre ai task,anche se bloccare gli interrupt per parecchio tempo può generare pasticci ancora più grandi.

Porte e Messaggi

La seconda possibilità si basa sull'intervento sincronizzato basato sulla tecnica dei segnali ed è di gran lunga la più consigliata. Molto semplicemente possiamo dire che un segnale indica l'arrivo di un generico messaggio proveniente da una qualunque altra parte del sistema. Un segnale è relativo al task che lo ha generato, un altro task non può manipolarlo, inoltre segnali identici possono avere significati differenti per ogni task.Praticamente per ottenere un segnale viene utilizzata la funzione AllocSignal(), ciò che ritorna è il valore numerico di quel segnale, mache può essere utilizzato solo dopo aver effettuato uno shift a sinistra.Le porte e i messaggi si integrano con i segnali e forniscono al programmatore un metodo per ottenere dell'intercomunicazione tra task. Le porte rappresentano ilpunto d'incontro di tutti i messaggi relativi ad un task.Exec definisce una porta attraverso la struct MsgPort:

```
struct MsgPort
   {
   struct Node mp_Node;
   UBYTE mp_Flags;
UBYTE mp_SigBit;
   struct Task *mp_SigTask;
   struct List mp_MsgList;
   }
```

Il campo mp_SigBit deve contenere il segnale generato dalla funzione AllocSignal(), perciò ad ogni porta corrisponde un segnale. Come nel caso della funzione AddTask() anche per le allocazioni delle porte i 3 programmi esempio usano una funzione di amiga.lib CreatePort() che provvede ad eseguire per noi le precedenti operazioni e qualche altra.Riguardo alle porte si deve menzionare il compito svolto dalla funzione FindPort.

Questa routine cerca semplicemente la struttura MsgPort corrispondente al nome passatole, ritornando un puntatore alla struttura stessa. Infine occorre ricordarsi che prima di eliminare una porta con RemPort() bisogna togliere tutti i messaggi accodati in quella porta.I messaggi rappresentano le informazioni che vogliamo passare tra i task, la corrispondente struct Message viene definita così:

```
struct Message
   {
   struct Node mn_Node;
   struct MsgPort *mn_ReplyPort;
   UWORD mn_Length;
   }
```

La struttura Message contiene un puntatore alla MsgPort in quanto il messaggio inviato da un task, può essere utilizzato dal mittente solo se il ricevente ha inviato un segnale sulla porta di risposta. Comunque i membri della struct Message non rappresentano alcun messaggio, perciò per ottenere lo scopo si deve allocare una struct Message dentro una struct definita dall'utente che contiene anche i messaggi da passare:

```
struct xxx
   {
   struct Message yyy;
   BYTE a,b;        I valori che interessano
   struct zzz *abc;  I il task ricevente
   }
```

Una volta definita la generica struct contenente i nostri dati per inviarla al task ricevente si utilizza la funzione PutMsg(mp,ms) dove mp è un puntatore alla MsgPort ricevente e ms è un puntatore alla generica struct.

Chiaramente il task ricevente dovrà prima o poi trovarsi nello stato diattesa, in modo da poter ricevere questo messaggio, ciò si ottiene utilizzando la funzione

Wait(). Gli argomenti di questa funzione sono i segnali allocati nelle porte interessate, quando uno di questi segnali arriverà il task uscirà dallo stato di attesa ed il programmatore, se vuole, può effettuare una scelta per sapere quale evento è occorso e conseguentemente scegliere.

I task non devono obbligatoriamente trovarsi in stato di attesa prima che venga eseguito l'invio del messaggio, infatti poichè i messaggi si accodano alle porte all'esecuzione della Wait() il task effettivamente non passerà allo stato di waiting bensì resterà nello stato di ready. Infine occorre far notare che aspettando per un unico segnale il programmatore può utilizzare la funzione WaitPort(mp) in alternativa alla Wait() dove mp è un puntatore alla MsgPort. Per raccogliere il messaggio si utilizza la funzione GetMsg(mp) dove mp è il puntatore alla MsgPort utilizzata. La funzione ritorna un puntatore alla generica struttura che contiene il messaggio interessato. L'ultima operazione da eseguire è la rispostache si ottiene tramite la funzione ReplyMsg(msg) dove msg è il puntatore alla generica struttura ritornato dalla funzione GetMsg().

Commento ai listati

Il nostro programma esempio (task) è composto da 3 sorgenti in C. Il primo (create.c) è il programma chiamante,cioè quello che permette di agganciargli altri 2 al sistema. Riguardo al file create.c è opportuno menzionarealcuni punti importanti: -la dichiarazione di tipo extern delle funzioni che rappresentano i task da aggiungere.
-NUMENTRIES rappresenta il numero di allocazioni richieste per ogni task, più precisamente stack pointer e struct Task.
-si rende necessaria l'istruzione #undef per ridefinire i #define senza provocare WARNING dal compilatore, tali #define allocano correttamente struct Task e stack pointer del secondo Task.

Durante le nostre prove abbiamo notato che il programma chiamante non può uscire prima che i task aggiunti non vengano rimossi, conseguentemente abbiamo utilizzato una variabile globale che viene settata solo all'arrivo di un messaggio di closewindow. Il controllo su questa variabile viene effettuato all'interno di un FOREVER. L'utilizzo di tale tecnica senza l'esecuzione di un minimo ciclo di attesa consuma molto tempo al processore, non permettendo nemmeno l'esecuzione di task con priorità negativa. Per evitare tale comportamento devastante abbiamo interposto all'interno del FOREVER una chiamata alla funzione Delay(timeout) che mette il task a riposo per il tempo definito da timeout (50 = 1secondo). Tale funzione è conforme alle regole multitasking del sistema e consuma un tempo veramente minimo del processore. Nei nostri esempi per ottenere il multitasking abbiamo utilizzato la funzione CreateTask(). Con questo metodo dopo una ovvia compilazione separata dei singoli file sorgente, segue un linkaggio globale dei 3 file oggetto, producendo un unico modulo eseguibile. Così facendo non avremo il prompt fino alla rimozione dei task, per ovviare a questo inconveniente dobbiamo lanciare il nostro programma mediante il comando RUN. Esiste un'altra possibilità di fare del multitasking, ciò consiste nel generare 2 file eseguibili mediante la sequenza separata compilazione-linkaggio, e agganciarli al sistema mediante il comando RUN del CLI senza utilizzare un file chiamante come nel metodo sopra indicato.

Il metodo principale per l'intercomunicazione tra task consiste nell'utlizzare la funzione FindPort() per la ricerca delle porte di comunicazione. Esiste un altro metodo che consiste nel dichiarare un puntatore alla porta sulla quale trasmettiamo di tipo esterno eseguendo poi una PutMsg() con quel puntatore, senza utilizzare la funzione FindPort(). Un' altra puntualizzazione importante sui due task aggiunti riguarda l'impossibilità di uscire liberamente mediante la funzione exit() perché a seguito di prove effettuate il sistema andava in GURU oppure ritornavadegli errori da parte del compilatore. Si è ovviato a tale inconveniente utilizzando un salto incondizionato ad una label posta al termine del programma. Nessuna particolare opzione di compilazione è stata utilizzata, per quanto riguarda il linkaggio eccovi il comando:

```
blink LIB:c.o+ram:create.o+ram:task1.o+ram:ta-
sk2.o TO ram:task LIB LIB:lc.lib+LIB:amiga.lib.
```

Conclusioni

Questo programma non vuole spiegare la differenza di utilizzo tra un task e un processo, ma solo spiegare un metodo di multitasking e relative intercomunicazioni. Come specificato all'inizio dell'articolo tutti e 3 i programmi sono stati sviluppati con il solo supporto del ROM KERNEL:EXEC, e mediante ripetute prove per individuare uno dei metodi corretti di funzionamento. Con questo speriamo di aver chiarito buona parte dei dubbi di come farsi del multitasking con Amiga.

Ed eccovi come promesso la funzione di amiga.lib CreateTask():

```
CreateTask(name,pri,initPC,stacksize)
    char *name;
    UBYTE pri;
    APTR initPC;
    ULONG stacksize;
{
    struct Task *newTask;
    ULONG memsize;
    struct FakeMemList fakememlist;
    struct MemList *ml;

    stacksize = (stacksize + 3) & ~3;

    fakememlist = TaskMemTemplate;
    fakememlist.fml_ME[ME_STACK].fme_Length
                                    = stacksize;

    ml = (struct MemList *) AllocEntry(&fake
                                    memlist);
    if (! ml) {
        return (NULL);
    }

    newTask = (struct Task *) ml->ml_ME[ME_TA
                                    SK].me_Addr;
    newTask->tc_SPLower = ml->ml_ME[ME_STA
                                    CK].me_Addr;
    newTask->tc_SPUpper = (APTR)((ULON
                                    G)(newTask-
>tc_SPLower)+
                            stacksize);
    newTask->tc_SPReg = newTask->tc_SPUp
                                    per;

    newTask->tc_Node.ln_Type = NT_TASK;
    newTask->tc_Node.ln_Pri = pri;
    newTask->tc_Node.ln_Name = name;
    NewList(&newTask->tc_MemEntry);
    AddHead(&newTask->tc_MemEntry,ml);

    AddTask(newTask,initPC,0);
    return(newTask);
}
```

E

ovvero

di Maura Moncaro

Parlare di una galleria d'arte su una rivista di informatica potrà sembrare strano a qualcuno; non certo però ai possessori di computer come Commodore Amiga. La ragione, per chi non lo sapesse(?!??!!), stà proprio nelle grosse potenzialità " artistiche " di Amiga.

Il motivo per parlare di EXIT, questo il nome della galleria d'arte, l'abbiamo colto -al volo- dopo il simpatico e interessante vernissage dell'esposizione dei lavori di Mauro Mauri; perche?

E qui c'è sempre lo zampino di Amiga.

Mauro Mauri è un artista che dopo varie esperienze, dal linguaggio figurativo iniziale alla nuova figurazione, al raffreddamento e alla geometrizzazione della pittura, fino a una pittura esterna allo spazio classico della tela -assemblaggio di materiali eterogenei, legno e tela con interventi pittorici di forte contrasto- e fino al lento recupero dell'immagine è approdato all'uso, come nuova tecnica grafica, del computer.

E qui di nuovo lo zampino di Amiga.

Le opere presentate alla galleria Exit ,da Mauri, sono state infatti prodotte usando semplicemente un Amiga 500 e una stampante Xerox 4020; per il pubblico, tra l'altro molto numeroso e particolarmente interessato, l'artista ha dato anche una dimostrazione pratica, all'inaugurazione e durante i giorni d'esposizione, di come si crea un lavoro artistico per tramite di un computer e di alcune sue periferiche.

Pensando con la testa di chi con il com-

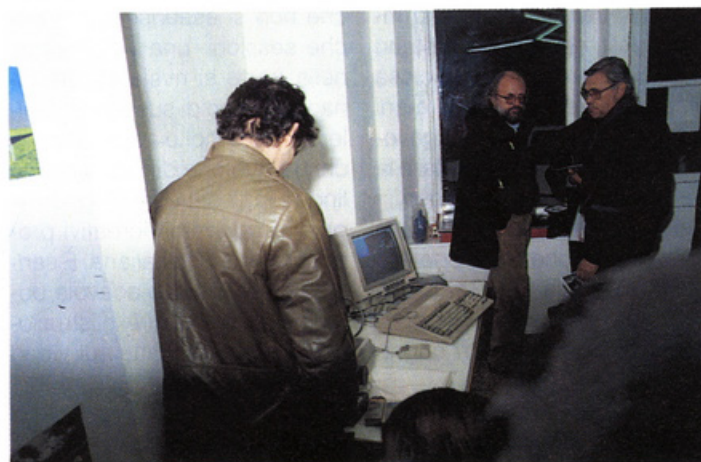
puter ha niente o ben poco da fare, è facile capirne la diffidenza verso una macchina che con l'arte dovrebbe aver poco da spartire; ma proprio rassegne artistiche come quella di Mauri, alla Exit, riescono a smuovere la diffidenza e lo scetticismo verso ciò che è diverso da una maniera comformista di vedere o fare le cose.

Ci è sembrato quindi giusto far conoscere ai lettori di questa rivista la volontà della galleria d'arte Exit a proporre nuove soluzioni e iniziative nel campo artistico, volontà che ,nella rassegna di Mauro Mauri, ha portato in galleria il loro amato Amiga.

Naturalmente chi ne è interessato,(artisti ,operatori del settore...) può contattare direttamente la galleria:

STUDIO D'ARTE " EXIT "
via Favetti 16\3
34170 GORIZIA
Tel. 0481-32396

Sequenza di foto del vernissage della mostra

TOP GAME

DRILLER

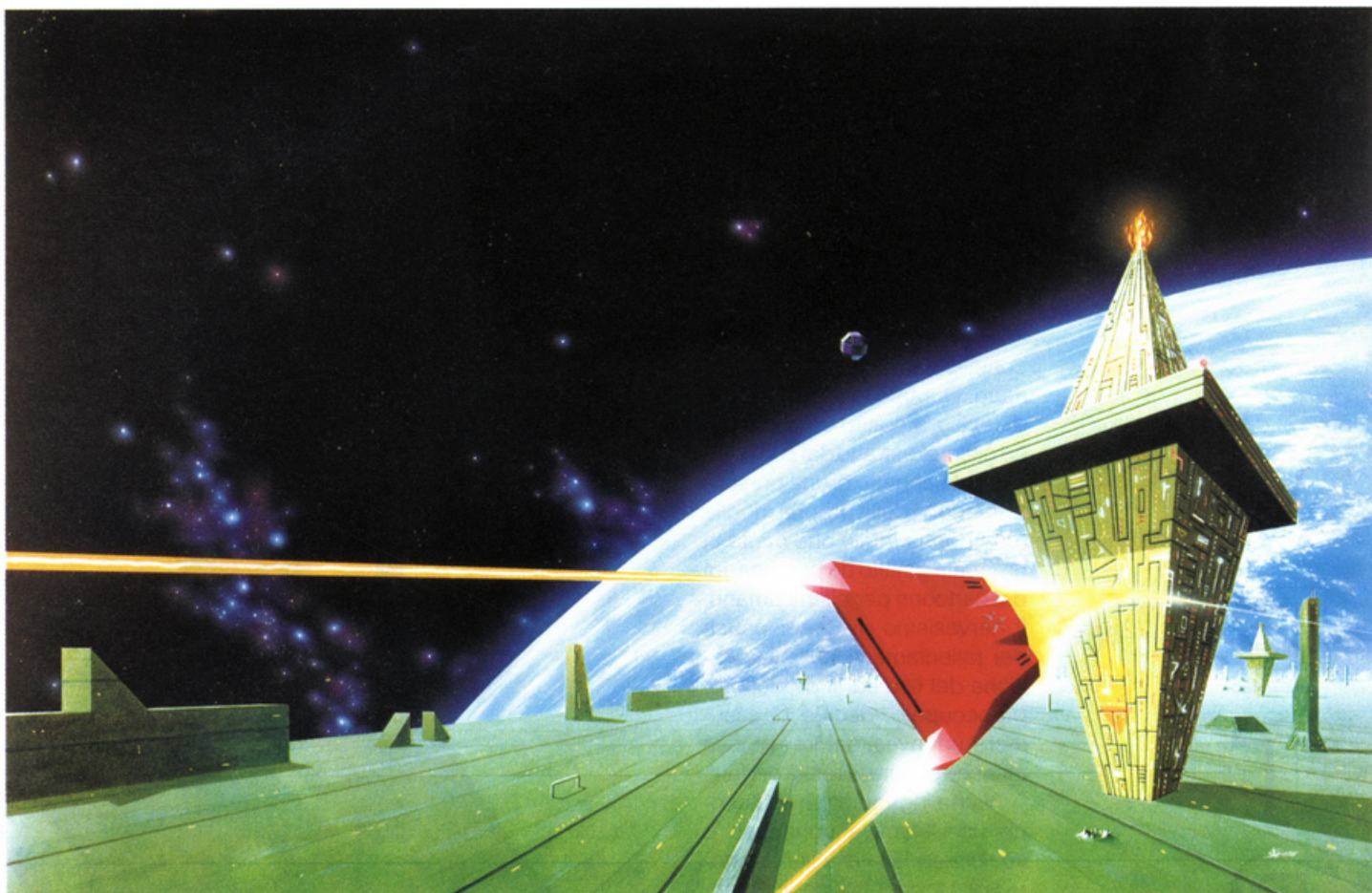
Demo version

Driller è il primo programma su Amiga a proporre l'innovativo sistema di visualizzazione dinamica denominato "Freescape". Questo metodo di programmazione consente infatti di muoversi in un ambiente completamente tridimensionale con un'assoluta libertà di movimento.

Sperimentato in precedenza sui Computer a 8 bit, Freescape ha raggiunto un livello qualitativo elevato solo sui Computer più potenti tra cui l'Amiga, che, grazie al processore di più recente tecnologia, consentono una velocità di scorrimento delle immagini ben più realistica ed efficace.

Siete su Mitral, una delle due lune di Evath. Gli Evathiani devono improvvisamente affrontare un gravissimo problema: una meteorite sta per raggiungere Mitral, che sotto l'inerme crosta nasconde un'enorme quantità di gas sotto pressione; la minima collisione può farla esplodere minacciando di mandare fuori orbita gli altri due pianeti del sistema, con conseguente morte certa per qualsiasi forma di vita esistente; L'unica possibile soluzione risiede nella trivellazione della superficie di Mitral nei punti in cui il gas è più concentrato, per farli evacuare almeno in parte e quindi mitigarne la compressione. La tua missione è molto ambiziosa, ma rappresenta anche la sola possibilità di salvezza per il pianeta: devi collocare un impianto di trivellazione in ognuno dei 18 settori in cui è stata geograficamente divisa la luna, localizzando le sacche e facendone uscire almeno la metà. Nel frattempo devi anche provvedere alle provviste energetiche per la tua sonda cercando particolari cristalli sparsi sul terreno di Mitral. Non mancano, infine, i soliti nemici che tentano di approfittare dell'incresciosa situazione per mettere le mani sull'ambito pianeta! Per agevolare l'orientamento ed evitare la perdita di tempo prezioso, i creatori di Driller hanno incluso nella confezione un modellino montabile della luna, in modo da poter indirizzare ogni spostamento con maggiori cognizioni.

Driller è senza dubbio un gioco d'azione, dinamico e brillante sotto l'aspetto estetico. In più, propone elementi strategici determinanti: l'abilità tattica del giocatore è tanto importante quanto la velocità di risposta in presenza di attacchi a sorpresa. Il controllo del veicolo a disposizione richiede inoltre di una certa familiarità con il pannello dei comandi, tramite cui si effettuano tutte le operazioni necessarie alle trivellazioni. Portare a termine la missione con esito positivo non è semplice: moltissimi elementi condizionano il successo delle proprie azioni, ed in ogni caso il tempo a disposizione è contato. Driller rientra in quella categoria di giochi che non si esauriscono in poche sessioni: una volta entrati nella parte si rivela appassionante e ricco di suspence. Driller è inoltre molto appetibile per chi ha qualche difficoltà con la lingua inglese: è infatti uno dei primi programmi ricreativi proposti in versione italiana. È senza dubbio molto piacevole poter finalmente capire le istruzioni e vedere le scritte sul video nella nostra lingua.

Buona Preview.

VETRINA

Chi ha incastrato Roger Rabbit, di Robert Zemeckis 1988

Roger Rabbit è un simpatico coniglio, divo del cinema animato. Durante la preparazione di un film il nostro eroe si trova ad essere distratto dal suo compito dal sospetto di una presunta infedeltà della propria moglie (una donna fatale che canta in un locale gestito da personaggi dei cartoons per clienti umani). Il nervosismo di Roger causa dei rallentamenti nella lavorazione del film ed il produttore, preoccupato, decide di assumere un investigatore privato affinchè scopra la realtà dei fatti.

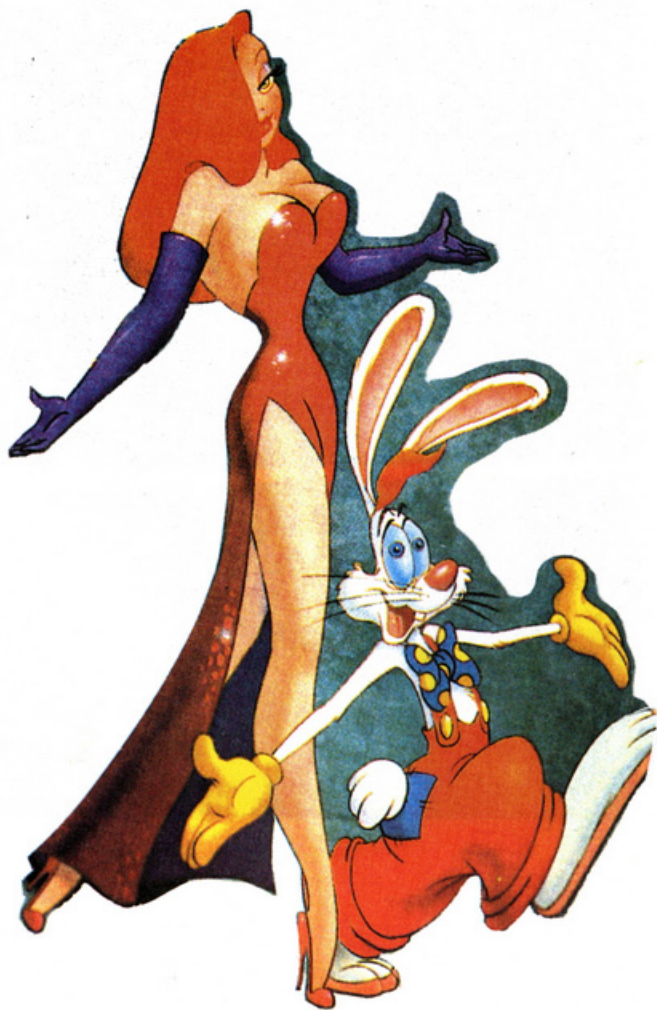
Viene ingaggiato a tale scopo un detective ubriacone e scalcinato: Eddie Valiant (l'attore Bob Hoskis). Dai pedinamenti effettuati dall'investigatore effettivamente sembra che la bella Jessica abbia una strana relazione con il proprietario di Cartunia (la città dove abitano i personaggi disegnati).

L'investigatore presenta le fotografie che dimostrano la colpevolezza di Jessica al povero Roger il quale, sconvolto dalla notizia ma ancor più dagli effetti strabilianti che hanno avuto su di lui pochi sorsi di liquore, fugge, non senza prima aver lasciato la sua sagoma sui vetri infranti della finestra chiusa dalla quale è uscito, nella notte per le strade di Hollywood. La stessa notte il presunto corteggiatore della affascinante moglie viene trovato ucciso e, ovviamente, Roger Rabbit viene immediatamente accusato dell'omicidio e, pertanto, ricercato dai crudeli sicari di un altrettanto crudele giudice. Roger, spaventatissimo, si rifugia nell'appartamento dell'investigatore Eddie il quale si trova, suo malgrado, coinvolto in una pericolosa quanto intricata storia di speculazioni edilizie e di tutti i più classici temi che si ritrovano in tutti i film polizieschi.

Come era facilmente prevedibile il nostro eroe e la sua bella moglie sono innocenti su tutta la linea ed insieme allo squinternato investigatore (che si rivela, per l'occasione, molto abile) dopo una serie di spassose, per lo spettatore, peripezie giungono alla scoperta ed all'eliminazione del vero colpevole salvandosi così dalla temuta "salamoia" (acido avente la funzione di "uccidere", sciogliendoli, i personaggi animati non distruggibili in nessuna altra maniera). Questa, a grandi linee, la trama sulla quale si innescano simpatiche e divertenti gag che movimentano tutto il film.

Il pregio di questo film non sta certamente nella trama che ripropone, volutamente, tutti i luoghi comuni dei classici hollywoodiani e della produzione di cartoni animati: l'investigatore ubriacone e squattrinato alla Bogart, la fatale Jessica che molto ricorda Rita Hayworth, il "cattivo" che, con i suoi sicari, perseguita la vittima del momento o il coniglio che racchiude in sè tutti i personaggi più famosi dei cartoni animati (la bocca simile a quella di gatto Silvestro, le orecchie di Willy Coyote, il papillon di Paperino prima maniera e gli occhi di Picchiatello), anche se tutti questi sono particolari per nulla casuali di un puzzle di soggetti, situazioni, gag quale è questo film.

Il vero grosso pregio del film "Chi ha incastrato Roger Rabbit" è da ricercarsi nelle modalità utilizzate per realizzarlo. L'in-

terazione tra personaggio umano e personaggio animato non è, di per sè, una grossa novità (tutti ricorderanno il balletto dei pinguini in Mary Poppins) mentre molto diversi sono i tempi di durata di questa interazione. Solitamente si trattava di pochi minuti, in questo caso si tratta, invece, del tempo di durata di tutto il film senza che mai venga a mancare la perfetta illusione che i personaggi animati siano reali anche se di un'altra specie. In questo film con la perfetta mescolanza tra attori e figure animate viene superata la barriera tra due mondi: quello della realtà e quello della fantasia.

Per giungere a questo risultato sono stati impiegati mezzi e capitali considerevoli, il film è stato tratto da un romanzo di Gary K. Wolf e diretto da Robert Zemeckis (regista di "Ritorno al futuro") e prodotto da Steven Spielberg. La parte animata è stata eseguita dagli studi Disney, le parti sono state disegnate interamente a mano senza l'ausilio di computer da più di 300 animatori diretti da Richard Williams (il disegnatore della Pantera Rosa) utilizzando la vecchia tecnica di Walt Disney (utilizzando cioè 24 disegni per ogni secondo di film a differenza dei 12 disegni utilizzati attualmente per ogni secondo di azione).

Sono state inoltre adottate tecniche particolarissime per ottenere alcuni degli effetti che tanto hanno meravigliato lo spettatore, per esempio per permettere ai personaggi animati di utilizzare oggetti reali (le pistole dei sicari del giudice, i vassoi dei camerieri, ecc.) sono stati impiegati dei burattinai che manovravano, durante la ripresa, gli oggetti o dall'alto con dei fili (come è stato fatto per le pistole) o con dei bastoni che reggevano i vassoi da sotto il pavimento (nel locale in cui cantava Jessica). Nel corso delle riprese, ovviamente, i personaggi animati non erano presenti ma venivano disegnati successivamente sulla pellicola in modo da coprire eventuali supporti esterni che comparivano inevitabilmente nel fotogramma.

Per permettere a Roger di bere tenendo in mano un bicchiere vero è stato utilizzato un braccio meccanico collegato ad un altro braccio indossato da un operatore. Al movimento di questo rispondeva il movimento corrispondente del primo, successivamente tale supporto veniva coperto dal disegno del personaggio. La preparazione tecnica di questa parte del film è durata circa due anni raggiungendo un costo da record: 250mila dollari al minuto. Molto bravi anche gli attori, specialmente Bob Hoskins che per cinque mesi si è trovato a recitare con interlocutori immaginari. A conti fatti bisogna render atto al regista R. Zemeckis e al disegnatore R. Williams di essere riusciti a mantenere la promessa fatta e cioè che Roger Rabbit sarebbe stata una sfida senza precedenti.

Immagine

In tema di revival interessante è da considerarsi questo doppio album di John Lennon intitolato "Imagine: John Lennon". Un titolo che oltre a ricordare quello che fu uno dei suoi brani più famosi vuole anche essere un ripristinare un'immagine del musicista che Lennon, sopra ogni altra cosa, è stato. La sua storia artistica ed umana fu varia: venuto alla ribalta come uno dei componenti del famosissimo gruppo dei Beatles lo abbandonò dopo alcuni anni sembra, a quanto ritenevano i suoi fans di allora, fortemente influenzato dal parere di sua moglie che fu ritenuta la causa principale dello scioglimento del gruppo. Dopo questa rottura Lennon si ripresentò al suo pubblico come solista dimostrando con le sue canzoni quell'impegno politico e soprattutto sociale che lo aveva sempre distinto.

La sua carriera finì, come tutti sanno, con un colpo di pistola che partì dalla mano di uno squilibrato mentale ponendo così fine alla vita di questo artista. Probabilmente questa tragica fine contribuì a crearne il mito anche se è innegabile il valore del contributo dato da questo artista nel restituire alla canzone popolare la sua funzione di veicolo di idee, siano esse politiche o estetiche. Innegabile è anche il fatto che tale messaggio fu recepito anche da milioni di giovani e non solo in quegli anni e che tale ripristino della antica funzione della canzone popolare influenzò, ed influenza tuttora, quasi tutti gli artisti rock venuti dopo i Beatles e dopo la sua morte.

Non vogliamo qui soffermarci su come fu come uomo anche perché ci pare troppo semplice osannare o dissacrare un mito quando ormai nessuno può più nè confermare nè smentire quanto viene detto (e qui mi riferisco chiaramente al libro di Goldman che è andato a rivangare in quella che è stata la vita privata di un uomo) ma vogliamo semplicemente ripresentare questo musicista la cui voce ed i cui ideali non sono mai tramontati e che pertanto mantengono intatta la loro freschezza.

Arredo pronto

L'ARREDO PRONTO proposto dalla Nicom è la risposta alle nuove esigenze dei consumatori. I mobili in questione vengono prodotti su scala industriale con materiali, tecnologie e design di alta qualità e sono confezionati, smontati, utilizzando tecnologie di assemblaggio che non richiedono nessuna esperienza di bricolage.

I vantaggi offerti da questi prodotti sono considerevoli: vendite non assistite da personale (il packaging è ampliamente autoesplicativo), disponibilità di cartellonistica promozionale, venduti senza bisogno di organizzare spazi espositivi e garanzia di soddisfazione del consumatore.

Ed è all'interno di quest'ottica che si inserisce il PC DESK offrendo una soluzione razionale per utilizzare in modo ottimale il proprio computer. I materiali impiegati sono dei laminati antigraffio di elevata qualità. Con PC DESK, la Nicom vi mette a disposizione una "stazione di lavoro" perfettamente organizzata e funzionale, in cui ogni cosa trova il suo posto e ogni cosa è disposta correttamente. PC DESK è composto di due elementi, disponibili in bianco e grigio, che possono essere acquistati separatamente.

L'ARREDO PRONTO ed il PC DESK sono prodotti dalla Nicom di San Paolo d'Argon (Bergamo), tel. 035/958224.

È IN EDICOLA

fare ELETTRONICA

N. 48 GIUGNO '89
L. 6000 - Frs. 9,00

Realizzazioni pratiche • TV Service • Radiantistica • Computer hardware

REALIZZAZIONI PRATICHE

Lampada da campeggio

Interfaccia MIDI per AMIGA

COMPUTER HARDWARE

Schede J-PC55 e J-I/O Card

Elettronica Facile

TELEFONO AMPLIFICATO

IONIZZATORE

RADIANTISTICA

Convertitore panoramico

TV SERVICE

Mivar T581

IN COLLABORAZIONE CON ETI ELECTRONICS TODAY INTERNATIONAL

GRUPPO EDITORIALE JACKSON
AREA CONSUMER

Taxe Percue (Tassa Riscossa) Milano Ferr. Corr.

Spedizione in Abb. Post. Gruppo III/70

"Fumetto in Computerart"
una produzione Graphic & Comp
Hardware: Amiga 2000
Polaroid Palette
Software: Deluxe Paint II

Felicita`
e`un pasto caldo

QUARTA PUNTATA

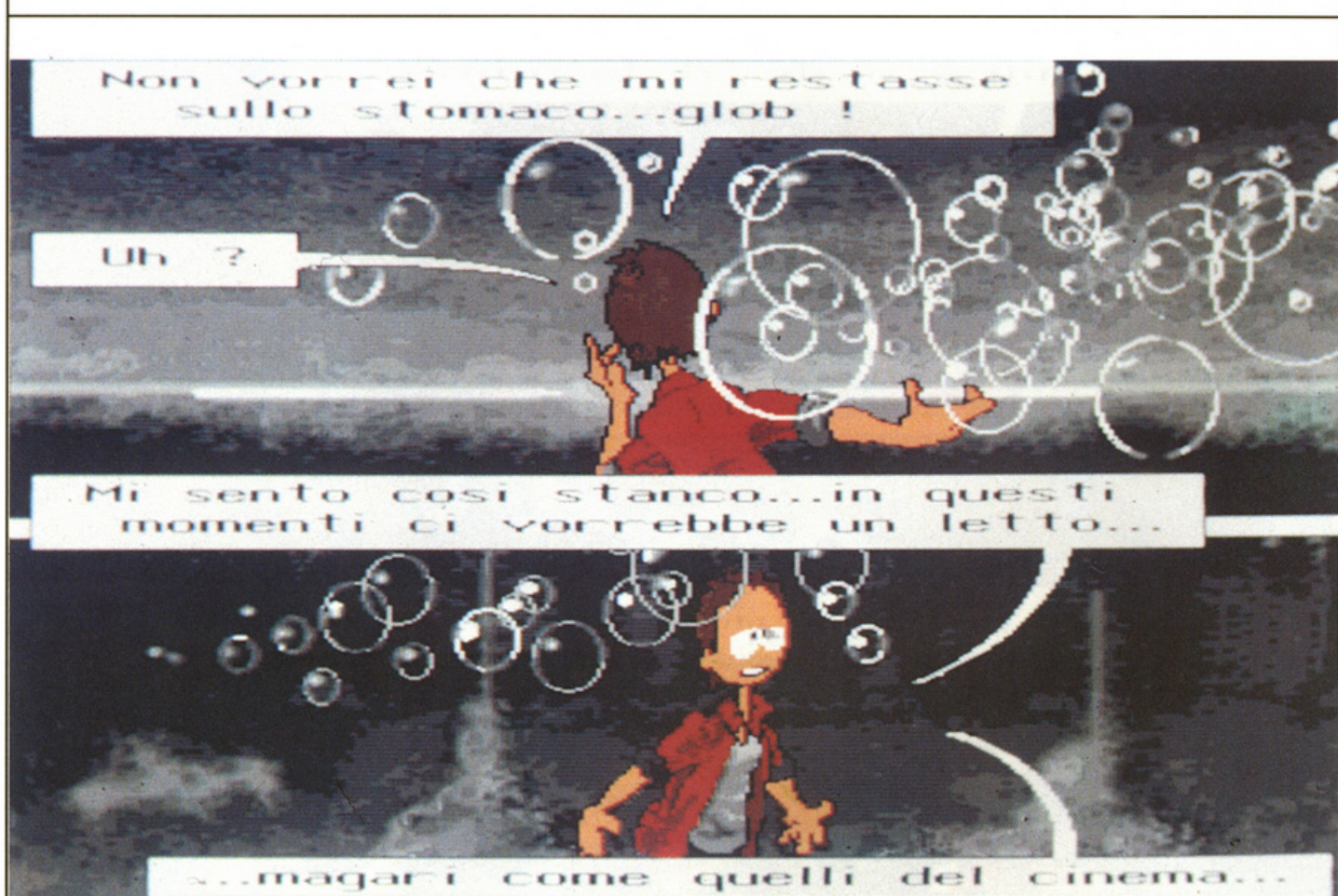
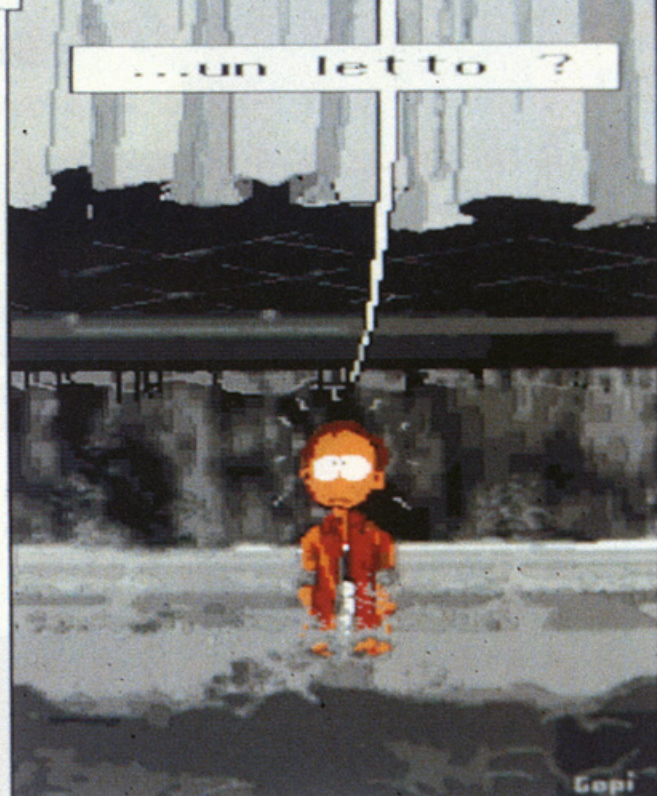
testo
e disegni
di GOPI

Che brutto colore ha,
chissa` il sapore...

GLOB!

WoW, mica male il brodino...

dammene ancora un goccio, figliolo

...uhm,Glom,però...
...yee, come mi gira la testa, non dovevo esagerare col brodo...
...una tazza forse bastava...
Hic!
Gopi
Non vorrei che mi restasse sullo stomaco...glob !
Uh ?
Mi sento così stanco...in questi momenti ci vorrebbe un letto...
...magari come quelli del cinema...

...o uno come questo ad esempio non sarebbe poi male !
Uh ? Ma cosa sto dicendo ?
...un letto ?
Gepi
URKA !! Un LETTO !
...e che letto !!
Dolce beatitudine...
Pevmette Eccellenza ?
Uh ? Chi e' ?

Il maggiordomo, Eccellenza...
Il "maggiordomo" ?
C'e' una visita per lei.
Ah, già e vero, io ho anche un maggiordomo Chi sarà mai a quest'ora ?

...una firmetta qui e gli Ippopotami non le faranno piu paura...
Fermo li !!
uh ?
Io le offro una cosa che ogni uomo ha sempre desiderato possedere...
Be` ,io...
...scommetto che indovina...
Non dia retta a quel tipo ! Ascolti me piuttosto...
la mia offerta non la potra' rifiutare !
Un fantastico campo da Golf !
...in sole novanta comode rate !
Fantastico !
Adoro il golf !
Mmm...
Si diverta amico mio...
Faro buca in tre colpi !

Che sball...
TUM
Hip!
Miseria, che botta...
Dove sono ?
Uh ?!
WOOOA
QUAW
...uh? Buio...gli indigeni... e il golf ?
Uuuh...il brodino, ora ricordo...
Ho sognato...
...uff...
infatti io odio il golf uh ?!
Ehi, chi mi palpeggia...

il ragnaccio !!
che faccio ora ?
...e...que...quello e'...
il MIO scafo !!
BIP BIP BIP
...se non fosse per il rumore della radio riuscirei a escogitare qualcosa...
la radio ?
la RADIO !!! Funziona ANCORA !!
URKA !

Non mi fai più paura ragnaccio...
...dammi la tua arma !
TU ! Brutto coso, lascia stare il mio scafo !
altrimenti...
PPPrrrrll-blub !
eh ?
Okay,okay...
L'hai voluta tu...
Via quel tentacolo !!
GGGRAWWLL...
TOC
wwoooouuuu...

CONTINUA

087P/C

Jackson

SOFTWARE & RIVISTE

D.O.C.

Scegli i giochi più freschi, quelli appena sfornati dal mercato internazionale. Poi, in edicola, chiedi le riviste più ricche di informazioni e idee per utilizzare il tuo computer nel modo più O.K. Ora unisci al tutto qualità, esperienza, tanta serietà, una lunga tradizione di fiducia e un pizzico di divertimento. Ecco fatto. Hai trovato il marchio di qualità che solo Jackson ti può offrire: è il marchio Jackson Software D.O.C. D'ora in poi cercalo sempre.

AREA CONSUMER

Scegli il meglio: scegli Jackson.

Compila e spedisci il coupon a: **GRUPPO EDITORIALE JACKSON AREA CONSUMER via Rosellini, 12 - 20123 MILANO**
Riceverai gratuitamente il nuovissimo adesivo "Jackson D.O.C."

SOFTWARE & RIVISTE D.O.C.

NOME ______ COGNOME ______
INDIRIZZZO ______ TEL. (   ) ______
CAP. ______ CITTÀ ______ PROV. ______
ETÀ ______ PROFESSIONE ______
QUALE COMPUTER POSSIEDI? ______
QUALI RIVISTE LEGGI? DI INFORMATICA ______
ALTRE ______
SEI ABBONATO AD UNA RIVISTA JACKSON? SI □ NO □
QUALE? ______
SEI TITOLARE JACKSON CARD? SI □ NO □
SILVER CARD N° ______ GOLD CARD N°. ______

Fondamentali per lo studio, il lavoro e l'aggiornamento

i dizionari enciclopedici di:

Eco

Se premete contemporaneamente i tasti ALT A potrete modificare il codice genetico in qualsiasi momento. Vi sarà risparmiato così il lavoro di trovare l'accoppiamento e quello della ricerca del cibo.

Barbarian

L'arco posto dietro il tinman è falso, ma potrete comunque prendere le frecce. Per eliminare il dragone non dovrete far altro che colpirlo con due frecce.

Insanity fight

Per portarsi alla schermata successiva si mantengano premuti entrambi i pulsanti del mouse, il tasto L ed infine si prema il pulsante di fuoco.

Test drive

Per migliorare la vostra guida nell'affrontare le curve di questo stagionato game, mantenete premuto il pulsante di fuoco quando la vostra macchina tende a slittare.

Hollywood poker

Per quei giocatori troppo intenti a giocare a carte, offriamo la possibilità di dare una prima occhiata alle schermate. Si inserisca Il dischetto contenente il Workbench e si premano i tasti CTRL D per portarsi in ambiente CLI, ora si digitino i seguenti comandi:

```
MAKEDIR RAM:
COPY C/CD RAM:C
COPY C/DIR RAM:C
COPY C/RENAME RAM:C
ASSIGN C: RAM:C
```

Si inserisca a questo punto il dischetto contenente il programma Hollywood Poker e si digiti quanto segue:

```
DIR
RENAME ISA.1 TEMP
```

Ora si scelga un file picture da analizzare. Il file avrà un nome di tre lettere seguito da un'estensione da 1 a 5, es: LOR.5. Prendete nota di quello che avete scelto e digitate:

```
RENAME  LOR.5 ISA.1
```

Effettuate il boot e poi premete il pulsante di fuoco per guardare la schermata.

Bard's Tale II

Anno di grazia 1219.

Aprofittiamo d'un momento di pausa nel susseguirsi delle avventure di Tangramayne per scrivervi questo pezzetto.

"Sono partito tranquillamente con un paio d'amici verso una nobile avventura in una lontana contrada. Si era in procinto di visitare una casa abbandonata quando il maggiordomo personale del Re ci sorprese e, con voce triste, iniziò un racconto."

"La figlia del Re è stata allevata da un certo Lord Dark che l'ha imprigionata nelle profondità d'un immenso dedalo che si trova sotto questa pianura. La sfida di Lord Dark è tremenda in quanto egli, sicuro della sua forza, non ostacola neanche la mia presenza nel suo regno sotterraneo. Il Re mi ha inviato alla ricerca di avventurieri disposti a rischiare la vita per salvare la Principessa."

"Senza esitare accettiamo immediatamente la missione; una lunga scala ci porta nella profonda oscurità del sotterraneo, dopo qualche passo veniamo attaccati da tre barbari, cinque maghi e tre orche. Ken liquida con le proprie mani un barbaro, Conan ne decapita un altro con la sua ascia di guerra e Arthur taglia le braccia al terzo. Io personalmente faccio appello alla magia per sopportare il soffio del drago e i maghi che mi stanno vicino. I denti aguzzi dell'orca addentano il braccio di Conan, guidato dalla rabbia e dal dolore egli trafigge l'orribile mostro. Ken dà il colpo mortale alla tempia dell'ultimo mostro, infine, dopo indescrivibili combattimenti scopriamo delle altre scale che conducono in tenebrosi abissi sotterranei. Prima di proseguire usciamo dal labirinto e ci sdraiamo sul prato per riposarci, curare le nostre ferite e lasciarvi delle indicazioni nel caso vorreste seguirci."

Ecco dunque il percorso a partire dal primo scalino:

Otto passi a destra (al secondo una voce vi parlerà); voltate a destra, fate tre passi; girate a sinistra, fate due passi; voltate a destra, fate un passo; infine a sinistra e fate undici passi (al settimo passo le vostre luci si accendono per magia).

"Noi ripartiamo per il secondo livello dopo essere passati dalla taverna e al tempio per sfamarci e metterci a posto con la coscienza..."

AMIGA GIOCHI

Star Fleet I

Avete appena concluso il vostro addestramento all'Accademia. Ora è giunto il momento in cui dovrete verificare se le innumerevoli ore trascorse al simulatore avranno un reale riscontro e l'unico modo per verificarlo è accettare il comando di una fra le più potenti navi da combattimento di tutto l'universo conosciuto, l'incrociatore pesante della classe INVINCIBILE. Il vostro compito è quello di proteggere le regioni esterne dell'Alleanza dall'invasione delle navi da guerra degli imperi di Zaldron e del perfido Krellan. Certi della vostra potenza vi immergerete nell'iperspazio sicuri del successo, e riemergerete nella regione di Deneb IV per ritrovarvi circondati dai distruttori di Krellan, ed ora... la guerra comincia!

"The War Begins!" è la prima di una serie di battaglie, azioni, programmi di strategie ideati con il concetto di FLEET. Indosserete i panni di un ufficiale di STAR FLEET e dovrete impegnarvi a fondo per avanzare dalla posizione di Cadetto di Accademia all'estremo rango di Ammiraglio. Migliorando le vostre abilità di comando otterrete delle missioni maggiormente impegnative e se la vostra guida risulterà essere impeccabile verrete insigniti della decorazione di Comandante di Navicella Spaziale o altre onorificenze ancora. I premi e le promozioni, conquistati con la vostra abilità, verranno registrati nel vostro registro delle missioni.

Le simulazioni delle azioni di battaglia sono pregne di entusiasmo e risaltano per le animazioni delle battaglie a colori, gli effetti sonori e la musica. Per giocare dovrete impostare delle strategie di raffinato livello dal momento che il nemico da sconfiggere oltre ad essere estremamente intelligente è anche perfidamente astuto.

Per comprendere meglio cosa stia accadendo nel venticinquesimo secolo seguiamo brevemente i pensieri del Capitano Wallace.

...l'ufficiale Wallace esce dall'ascensore, dirigendosi verso il ponte, e saluta con un cenno la guardia alla porta. I suoi occhi osservano il ponte, mentre viene notato a malapena da un gruppetto di uomini intenti nell'espletamento dei loro compiti. Al timone vi sono due nuovi ufficiali, appena usciti dall'Accademia. Il capitano Wallace scuote leggermente la testa e cerca di ricordare i nomi dei 50 nuovi uomini che sono stati assegnati alla Saratoga, la navetta dell'Alleanza Galattica, nell'ultimo mese. Questo è un triste segno del prezzo che la guerra sta facendo pagare alle forze dell'Alleanza.

Appena seduto, si rivolge alla sua destra, dove il Primo Ufficiale sta controllando i sensori, e commenta: "Certo che ora è calmo". "Sissignore" risponde il Primo Ufficiale senza distogliere lo sguardo dalla consolle. Ultimamente i compiti di chi sta sul ponte sono così, specialmente per la pattuglia della Zona Neutrale. Timore e tensione gravano sul morale della truppa.

Passando le dita tra i capelli divenuti precocemente grigi, il Capitano Wallace ripensa a quando era un giovane ufficiale. Mentre si siede al posto di comando, ripassa mentalmente i giorni avventurosi delle esplorazioni, in tempo di pace, prima che scoppiasse la Seconda Guerra Galattica, che ancora oggi continua. Ripensa a quando fu assegnato alla navetta U.G.A.S. Hornet ed ai due anni di esplorazione nella regione di Archenar, tanto tempo prima dell'insurrezione dell'Impero di Krellan.

Dopo la Prima Guerra Galattica, l'Impero dei Krellan era tutto fuorché estinto. I Krellan erano stati limitati, grazie ad un trattato di pace, sia nel territorio che nella forza militare, il che non si adattava molto alla loro natura barbarica. Poi essi elessero un nuovo imperatore: Henry Zae IV.

L'imperatore dimostrò ben presto di essere il genio politico del XXV secolo. Sotto il suo dominio l'Impero Krellan iniziò rapidamente a prosperare. Ma ancora più importante è il fatto che quella gente così brutale riguadagnò la potenza militare e l'orgoglio grazie al nuovo imperatore. Dopo due anni che era in carica, l'Impero Krellan occupò una zona neutrale al di fuori dell'Alleanza, senza sparare un colpo.

I Senatori dell'Alleanza prestarono però poca attenzione

all'espansione dell'Impero dei Krellan. Il popolo Krellan iniziò ben presto ad adorare Zae come il suo Messia, mentre l'Impero continuava a crescere. Propaganda e pregiudizi in breve si diffusero per l'intera galassia. Durante un discorso interstellare, Zae discusse della superiorità dei Krellan e affermò che la Razza Dominante Barbarica era stata creata per comandare. Gli storici paragonarono Zae a Zaldred della Mecca, a Hitler della Terra ed a Estar di Vega. Tutti questi Messia auto-proclamati portarono sempre guerra e distruzione, nonostante ciò i leader dell'Alleanza non vi prestarono nuovamente molta attenzione.

Poco dopo il discorso di Zae, la sua intelligenza cominciò a manifestarsi. Nell'Alleanza circolavano voci che Zae avesse stipulato un trattato con la gente di Zaldron, una razza aliena imperialista. "Impossibile", proclamarono i leader del Senato. "Gli Zaldron non comprometterebbero mai la sicurezza dei loro mondi per un fanatico come Zae." "Assolutamente incredibile," continuavano a dire.

...ma la mattina del 3095.7 (Data Galattica; 29 agosto 2414 D.C., calendario solare) la Galassia fu costretta a prestarvi attenzione. Gli Zaldron attaccarono un piccolo pianeta militare nei dintorni di Deneb IV. Questo pianeta era Beta II, un Comando Regionale della Flotta dell'Alleanza. Le forze d'occupazione Zaldron avevano fatto irruzione nel territorio dell'Alleanza, passando completamente inosservate, ciò fu possibile grazie all'utilizzo di schermi invisibili che, fortunatamente, Zae non possiede ancora. Dopo che gli Zaldron ebbero distrutto le difese di Beta II, la flotta dei Krellan diede inizio all'invasione.

Il trattato siglato dalle due potenze barbariche ebbe come risultato il massacro di Beta II. L'attacco a sorpresa provocò la morte di oltre 94.000 uomini dell'Alleanza, il pianeta non era più abitabile e la flotta dell'Alleanza era andata in gran parte distrutta. Questo fu l'inizio della cruenta Seconda Guerra Galattica.

Il violento attacco portato dai Krellan non terminò qui. Altre basi dell'Alleanza, che era scarsa, ed impreparata, fu presto spazzata via. In poco tempo all'Alleanza rimasero solo una manciata di navi da guerra e, nelle regioni più esterne, si ritrovarono indifesi e senza aiuto.

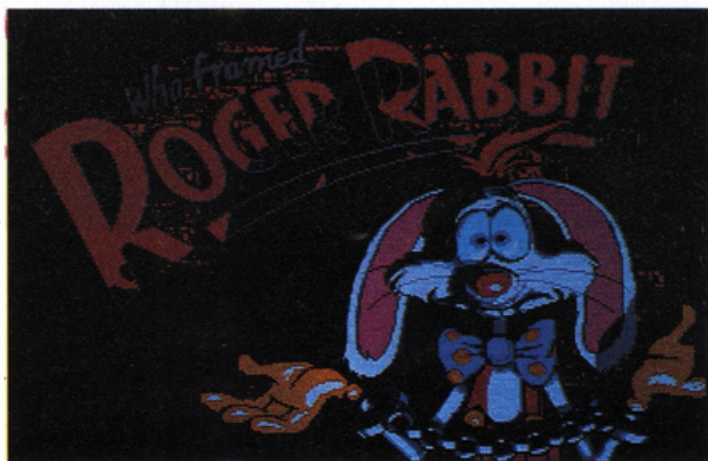
Le truppe, abituate fino a quel momento alle esplorazioni spaziali, si ritrovarono all'improvviso nel pieno della guerra. All'Alleanza si presentò la scelta obbligata di richiamare le navi addette all'esplorazione per poterle adibire alla difesa delle regioni più esterne. A queste navi solitarie fu affidato il doppio compito di proteggere queste popolazioni abbandonate e lontane e contemporaneamente di guadagnare tempo a favore dell'Alleanza in modo da poter rimettere in sesto la flotta.

Improvvisamente i pensieri del Capitano furono interrotti da un suono familiare. La sirena d'allarme stava scuotendo la nave. Varie navi spaziali erano in avvicinamento: ...veloci, ...nemiche, ...KRELLAN! "ALLARME ROSSO - QUARTIERE GENERALE!"

Questo gioco di guerra strategico si svolge tra due opposte fazioni. Le flotte imperiali di Krellan e di Zaldron stanno mettendo in pericolo la democrazia, e dovrete fermarle ad ogni costo. Durante il gioco incontrerete navette spaziali nemiche che vi combatteranno, clandestini che saboteranno la vostra nave, ed altri inconvenienti ancora. Inoltre la Interstel ha incorporato in Star Fleet I una caratteristica che lo distingue dagli altri giochi di guerra spaziali, e che motiva il suo nome. Diventerete un membro di Star Fleet, e potrete misurarvi contro i vostri amici cercando di progredire attraverso i vari gradi, da semplice Cadetto dell'Accademia degli Ufficiali di Star Fleet ad Ammiraglio. Sono stati scelti alcuni standard minimi che occorre superare per essere promossi. Di ogni giocatore verranno registrati i dati storici e quelli relativi alle promozioni ottenute. Questi dati potranno essere letti da tutti i membri della flotta. L'obiettivo finale è di raggiungere il grado più alto, che è quello di Ammiraglio di Merito; il fatto di poter vedere i progressi degli altri giocatori rende la competizione più agguerrita ed avvincente.

Per spingervi a dare il massimo nelle missioni individuali sono stati previsti dei premi e delle decorazioni. Questi verranno attribuiti automaticamente dal programma a seconda dei risultati conseguiti nelle missioni individuali e saranno immesse nel vostro registro personale delle missioni, in questo modo chiunque potrà visualizzarle.

All'inizio, come cadetto dell'Accademia, comanderete la navetta di arruolamento Republic. Superata questa fase avrete a disposizione una flotta composta da trentasei navi spaziali. La maggior parte di queste hanno il nome di famose navi da guerra del passato (Apollo, Atlantis, Bismark, Eldorado, Saratoga, Yamato, Nimitz, ecc.); potrete, se lo desiderate, comandare una nave diversa ad ogni nuova avventura. Non dovrete far altro che sistemarvi al posto di comando e prepararvi ad affrontare un eccitante viag-

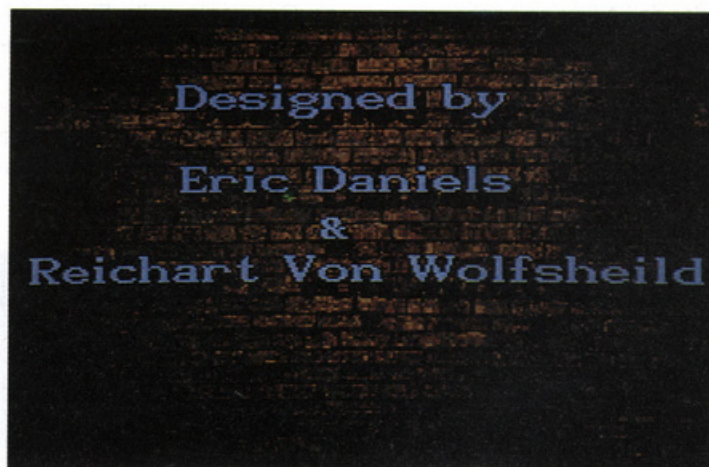
gio nel vuoto interstellare.

Anche se il manuale è consistente vi consigliamo di leggerlo attentamente prima di iniziare il gioco. Ci sono molte cose che un comandante spaziale è tenuto a conoscere ed il programma non contiene alcun tipo di istruzione.

Per quanto riguarda il nemico vi ricordiamo che esistono due razze di alieni ostili: i Krellan e gli Zaldron. Il Servizio di Spionaggio di Star Fleet è riuscito ad identificare un solo tipo di astronave Krellan: il Distruttore. In merito alle navi utilizzate dagli Zaldron non siamo in grado di fornirvi alcun elemento nè sulla classe nè sul tipo, possiamo solamente anticiparvi che le dimensioni sono simili a quelle del Distruttore.

La popolazione Krellan viene classificata come umanoide a sangue caldo ed è una razza estremamente ostile ed aggressiva. Sono dotati di forza fisica superiore a quella umana ed alla maggioranza degli abitanti della Galassia. La secolare selezione razziale li ha dotati di un comportamento aggressivo e di un coraggio che potremmo definire selvaggio. Il principio su cui si basa la loro società è la conquista. I piccoli dei Krellan vengono selezionati per una futura carriera militare. La massima aspirazione dei genitori è quella di sperare che i loro figli vengano inviati a far parte delle truppe d'assalto dell'Impero, altrimenti dovranno rassegnarsi a vederli crescere nella classe operaia dove lavoreranno tutta la vita per aiutare l'Impero. Tutte le cariche politiche, mediche o professionali sono generalmente affidate agli ufficiali imperiali. Lo scopo fondamentale nella vita di un Krellan è la guerra e le conquiste. Un vecchio Krellan soleva affermare "Vivere per conquistare e conquistare per vivere!"

Gli Zaldron sono dei rettili a sangue freddo e abitano nel sottosuolo. Questi esseri sono forniti di un notevole livello di intelligenza e non sono aggressivi come i Krellan. La caratteristica principale di questa razza è la pignoleria, infatti se si analizzano le loro battaglie si può notare che sono state tutte meticolosamente studiate prima di essere eseguite. Questo popolo è guidato da una regina e tutti i

posti di responsabilità, sia politici che in qualsiasi campo professionale, vengono occupati dalle femmine. In questo modo i maschi si occupano esclusivamente della guerra, coloro che non sono adatti a questo compito diventano immancabilmente aiutanti degli ufficiali femmine. In un primo momento gli Zaldron auspicavano una possibile alleanza con gli Stati dell'Alleanza, ma le ambizioni espansionistiche della loro regina portarono invece ad un patto con l'Impero Krellan.

Ricordatevi che la tattica adottata dai nemici dipende dal livello della vostra missione. Nel primo livello non si muoveranno assolutamente, mentre nel secondo livello si sposteranno solamente all'interno del quadrante utilizzato in quel momento. A partire dal terzo livello le operazioni si faranno più complesse e i nemici si sposteranno, oltre che dentro il quadrante, anche da quadrante a quadrante all'interno della regione. Sarete comunque in grado di osservare i loro movimenti all'interno dei quadranti esplorati della vostra regione tramite i sensori ad ampio raggio ed osservando la mappa della regione. Utilizzando la mappa della regione sarete in grado di osservare se una base spaziale sta per venire attaccata o se è preferibile mutare tattica e navigare verso altre regioni alla ricerca di altre forze aliene.

Una delle strategie preferite dai Krellan è quella di concentrarsi vicino ai quadranti dove sono poste le basi spaziali dell'Alleanza Galattica e per ogni quadrante ci potranno essere fino ad un massimo di cinque Krellan. Per quanto riguarda le navi Zaldron queste possono entrare ed uscire da un quadrante in qualsiasi momento e operano individualmente. Le navi Zaldron risulteranno invisibili grazie a degli speciali schermi di cui sono dotate. Comunque la loro presenza potrà essere rilevata per mezzo di sensori che sono in grado di scoprire se ci siano dei disturbi nella dimensione spazio - tempo, ma solo casualmente riuscirete a rilevare la loro esatta posizione.

Va sottolineato il fatto che gli schermi utilizzati dagli Zaldron consumano moltissima energia ed è difficile mantenerli stabili. I vostri sensori a corto raggio saranno perciò in grado di rilevare dei disturbi che si manifestano con dei tremolii del visualizzatore tattico del settore che contiene la nave Zaldron. Dal momento che gli Zaldron si servono di moltissima energia per rimanere invisibili, se rimarranno per periodi troppo lunghi nel vostro quadrante, diverranno troppo deboli finendo col divenire visibili e non riuscendo più a muoversi.

Potrete essere colpiti da navi nemiche ogni volta che accade uno dei seguenti fatti:

1 - Quando entrate in un nuovo

quadrante.
2 - Quando vi muovete.
3 - Quando azionate siluri o phaser
4 - Quando utilizzate l'argano o i trasportatori.

Se lascerete trascorrere un'intera giornata senza che il nemico apra il fuoco nei vostri confronti, esso vi sparerà comunque. Più il nemico è vicino e più potente sarà il colpo. Quando danneggiate una nave nemica, essa perderà notevole potenza. Non dovrete eliminare tutti i nemici presenti nella vostra regione, dal momento che le navi aliene saranno sempre maggiori di quelle necessarie per portare a termine la vostra missione.

Dovrete far particolare attenzione agli intrusi dal momento che questi non sono altro che degli alieni che salgono a bordo della vostra navetta con il preciso scopo di distruggere i sistemi operativi. Gli intrusi possono essere agenti Krellan o Zaldron trasportati a bordo attraverso uno scudo di difesa spento o agenti Krellan introdottisi mentre siete in fase d'attracco con la base spaziale o infine dei prigionieri evasi.

A bordo potrà esserci solamente un intruso alla volta e viene localizzato solamente quando sta sabotando il sistema operativo della navetta.

Sarete eliminati dal gioco se la vostra navetta verrà messa fuori uso o distrutta, il che potrà accadere in uno dei seguenti modi:

- Termine dell'energia.
- Distruzione dei supporti vitali Primary e Backup.
- Esaurimento delle batterie del sistema di supporto Backup.
- Morte dell'intero equipaggio che si trovi a bordo della navetta.

Le basi spaziali vi saranno utilissime per i rifornimenti e le riparazioni. Queste basi sono sparse nello spazio controllato dall'Alleanza e servono oltre che per il rifornimento delle navette anche per preparare un primo fronte di difesa in caso di invasioni aliene. Dopo essere attraccati verranno riparati tutti i sistemi danneggiati, rifornita la navetta di potenza, le truppe ed i siluri verranno sostituiti. Durante la fase di attracco la base spaziale fornisce una forza di sicurezza che vi protegge se avete un intruso a bordo mentre siete attraccati. Se durante il periodo in cui voi siete attraccati la base viene attaccata verrete distrutti insieme alla base.

Promozioni, premi e decorazioni verranno trascritti nel Registro delle Missioni, che verrà aggiornato automaticamente dal programma al termine di ogni gioco.

Per mantenere queste ed altre informazioni STAR FLEET I utilizza tre file di dati inseriti nel dischetto. Dopo il completamento di ogni missione questi file vengono aggiornati automaticamente. Se malaugauratamente avete cancellato un file o all'interno del dischetto ci sono dei settori rovinati la Interstel ha provveduto ad inserire nel programma degli strumenti per manipolarli.

Per concludere vi riportiamo una parte del discorso del comandante ed alcune informazioni tecniche e tattiche.

...Benvenuti a bordo! Desidero congratularmi con voi per essere stati accettati all'Accademia degli Ufficiali di Star Fleet. Qui riceverete le basi per comandare le più sofisticate navette spaziali dell'universo conosciuto. Dopo aver completato gli studi in classe, sarete assegnati alla nave - scuola U.G.A.S. REPUBLIC.

Quando avrete terminato la preparazione, ricoprirete il ruolo di ufficiale nella più potente forza di pace della galassia. Spero che otterrete un ruolo di vostro gradimento.

Da parte dell'Accademia, spero che il soggiorno presso di noi sia di vostro gradimento.

Distinti saluti,
Vice Ammiraglio Robert L. Winkler
Comandante, Accademia degli Ufficiali di Star Fleet

Il quartier generale del Comando di Star Fleet è situato sul pianeta Cygni Epsilon Three. Il pianeta fa parte di un gruppo di quattro pianeti abitabili che gravitano attorno alla stella di Gienah, posta sulla parte sinistra della croce immaginaria formata dalla costellazione Cygnus. A causa della forma del gruppo di stelle di Cygnus, essa fu fu chiamata dagli astronomi terrestri Croce del Nord.

A questo punto passiamo al gioco vero e proprio, il quadrante che vi verrà assegnato nei primi due livelli è privo della presenza di navi nemiche, potrete perciò familiarizzarvi e provare allo stesso tempo i comandi. Non fatevi scappare quest'occasione dal momento che nei livelli superiori le cose diventeranno estremamente complicate. Ci sono inoltre alcune semplici regole che si dovranno seguire per non incorrere in banali errori:

Le coordinate sono indicate come riga, colonna ed il loro conteggio inizia dal bordo superiore sinistro del quadrante. Ricordatevi che il C-Factor rappresenta una distanza. Le voci course, heading, bearing hanno lo stesso significato. Queste parole descrivono la direzione del movimento o la direzione della vostra navetta verso un certo bersaglio.

La vostra navetta è protetta da quattro schermi di energia indipendenti tra loro; questi schermi assorbono e dissipano l'energia sprigionata dalle armi nemiche. Il primo schermo protegge la parte frontale mentre gli altri scudi vi proteggeranno a babordo, poppa e tribordo.

Se entrate in collisione con un oggetto, stelle, basi spaziali o altro, la navetta sarà fermata automaticamente dal computer di navigazione in tempo per evitare danni. Se entrate in collisione con un'astronave nemica, il computer vi aiuterà a prevenire i danni risultanti da una collisione. Quando viaggiate nell'iperspazio, la vostra navetta non subirà collisioni ma sarà soggetta alle tempeste ioniche.

I sistemi di bordo danneggiati verranno riparati dagli addetti e dal computer. Le giornate richieste per la riparazione prendono il nome di Estimated Repair Time e sono visualizzate nel bollettino della Situazione dei Danni. La rapidità delle riparazioni è direttamente dipendente dal numero dei soldati a bordo della navicella e dalle condizioni di allarme del mezzo stesso.

La navetta è fornita di tre sonde spaziali di riconoscimento, che vi verranno utili al momento di esplorare i quadranti limitrofi. Per un utilizzo ottimale delle sonde dovrete avvicinarvi al bordo della regione e lanciarle con una angolazione tale che permetta di effettuare il percorso più lungo. Queste sonde non vengono fornite dalle basi spaziali e se attraversano un quadrante ostile, c'è la possibilità che i Krellan le distruggano.

Se malaugaratamente avete un sistema di supporto vitale danneggiato cercate di non lasciarlo a lungo fuori uso. La navetta è provvista di un secondo supporto vitale che sostituisce quello primario quando viene danneggiato ma se subite un ulteriore attacco che danneggi anche il sistema secondario verrete uccisi assieme al vostro equipaggio e la navetta risulterà essere fuori uso.

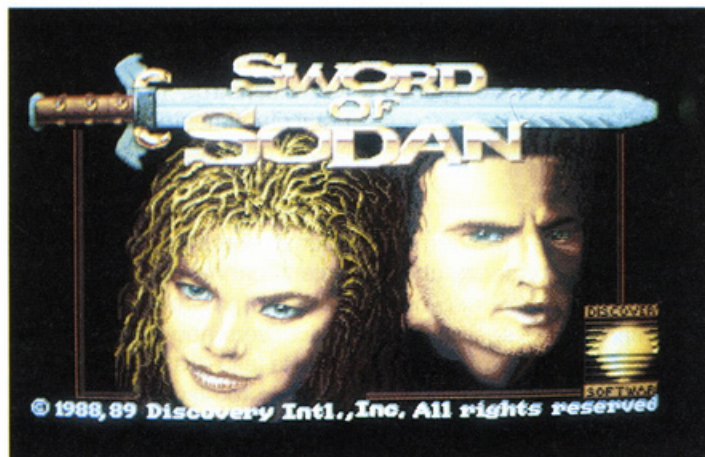
Dopo le prime esperienze di volo dovrete acquisire le conoscenze più elementari riguardo le strategie e le tattiche da utilizzarsi nelle future missioni. In qualsiasi guerra, il vincitore riesce a trionfare grazie all'acquisizione di qualche vantaggio sull'avversario. Il vantaggio può essere costituito da armi, truppe, equipaggiamento delle forze, spionaggio, ecc. STAR FLEET I non fa eccezione a questa regola.

Nel Manuale degli Ufficiali rintraccerete la descrizione delle armi, dei sistemi di bordo, delle mine e degli altri strumenti con cui attaccare gli alieni. Se non conoscerete perfettamente queste armi renderete inutile anche l'arma più pericolosa. L'abilità con cui utilizzerete le armi a vostra disposizione sarà la chiave che vi permetterà una lunga carriera piena di successi come ufficiale di STAR FLEET.

Per combattere gli Zaldron nelle loro navette invisibili si possono utilizzare diverse tattiche. La prima tattica è la più efficace ai livelli inferiori. e consiste nel farsi sparare (muovendosi) e nell'osservare quale sia lo scudo colpito, in questo modo saprete la direzione in cui lanciare dei siluri in modo manuale. Aumentando di grado, questa tattica non sarà più utilizzabile, dal momento che aumenteranno le possibilità che la navicella Zaldron si sposti immediatamente dopo avervi sparato. Non utilizzate questa tattica quando siete a corto di energia.

La seconda tattica è molto

simile alla prima, unica differenza è che vengono utilizzati i phaser al posto dei siluri. Questo metodo presenta gli stessi limiti di quello precedente.

L'ultima tattica si serve dei siluri e delle mine per catturare l'alieno. Questo metodo è efficacissimo ai livelli più alti del gioco e permette di depositare una striscia di mine disposta attraverso il quadrante. Non scordatevi che una volta lasciato il quadrante, le mine depositate andranno perse. Le mine inesplose si possono recuperare per rifornire la dotazione di siluri.

Le basi spaziali vi serviranno per il rifornimento di energia e per la riparazione della navetta. Quando attraccate verranno sostituiti i membri dell'equipaggio persi durante la missione. Per attraccare ad una base dovrete porvi in uno degli otto settori che la circondano e immediatamente il computer vi chiederà se volete attraccare. La navetta verrà rifornita solamente una volta per ogni attracco. Le basi rigenerano lentamente le proprie energie, l'energia viene prodotta in misura del numero delle stelle presenti nel quadrante.

La tattica migliore è quella di trovare una base prima di averne bisogno. La collocazione delle basi è tenuta segreta per ragioni di sicurezza, perciò le dovrete trovare senza alcun aiuto. Nei primi gradi le basi sono in un numero maggiore e sono facili da trovare. Ma passando di grado in grado le basi diminuiscono. Agli ultimi due gradi esiste solamente una base. Potrete lanciare una sonda e utilizzare i sensori a lungo raggio per localizzare una base. Se non siete in grado di eseguire il rifornimento o le eventuali riparazioni, è preferibile evitare lo scontro con il nemico.

Dopo aver disattivato una navetta aliena, potete utilizzare i marines spaziali per abbordarla e catturarla. Si potranno catturare i prigionieri e imbarcarli sulla propria navetta. Dopo aver posto in posizione di rimorchio la navetta aliena, dovrete portarla ad una base e consegnarla alle autorità dell'Alleanza. In alcune situazioni potrete trovarvi in un quadrante nel quale parecchie navi aliene siano state catturate e poichè possedete un solo argano, potrete rimorchiare solo una navetta per volta. Se abbandonate un quadrante dove è presente una nave aliena fuori uso quando ritornerete la troverete completamen-

te riparata e con il massimo di energia. Perciò non lasciate mai una nave fuori uso in un quadrante e se vi trovate in una situazione simile a quella che vi abbiamo esposto cercate di rimorchiarne una e distruggete il resto della flotta.

Il quadro che emerge da un'analisi dei viaggi spaziali non può definirsi completo se non vengono citati almeno per sommi capi alcuni rischi che si possono verificare durante questi spostamenti interstellari.

Le esplorazioni dello spazio a scopo pacifico non sono occupazioni prive di rischio. I comandanti delle astronavi hanno scoperto varie forme di vita, alcune delle quali sono state riconosciute nocive per l'uomo. Molti comandanti hanno evidenziato uno strano, inspiegabile comportamento durante una missione. Ecco due delle più comuni malattie.

La Manovra di Nobles - Questa malattia fu chiamata con il nome del Vice Ammiraglio Charles Nobles. Il sintomo più comune è l'impossibilità del comandante di pilotare la navetta. Un esempio tipico è quello di chiedere al Controllo di Navigazione un C-Factor di 13 invece di 1.3. Il Vice Ammiraglio Nobles ed il suo equipaggio diedero le loro ultime notizie durante un viaggio nella regione di Antares VI.

La Sindrome di Tomodachi - Questa malattia fu chiamata con il nome del Commodoro "Bonzai" Tomodachi. Il sintomo più comune è l'impossibilità di essere promossi senza difficoltà da alcuni gradi ad altri, mentre con altri non vi è alcun problema. Per esempio, il Luogotenente Tomodachi fu promosso Comandante Luogotenente dopo 14 missioni; ne impiegò 5 per diventare poi Comandante, altre 32 per il grado di Capitano e finalmente diventò Commodoro dopo otto missioni. Il Commodoro Tomodachi deve ancora essere promosso Ammiraglio in Seconda dopo aver completato 57 missioni come Commodoro. Benchè il virus che causa tale malattia debba ancora essere isolato, il Comando di Star Fleet sta studiandolo con impegno, poichè la salute degli ufficiali è di primaria importanza. La sindrome di "Brick Wall" è simile alla Sindrome di Tomodachi, ma invece di essere ciclica, essa è come un muro impenetrabile e quindi non si ottengono più promozioni.

Il viaggiare attraverso le grandi distanze dell'Universo comporta notevoli pericoli. Vi sono molti fenomeni inspiegati e sconosciuti nell'Universo. Uno di questi è la tempesta ionica. Sebbene molte delle cause di tali tempeste siano conosciute (esplosioni di nova, nebulose, disturbi gravitazionali, buchi neri, warmhole, ecc.), la loro dislocazione è casuale e non prevedibile. Alcune di queste tempeste sono state seguite per anni luce dalle navette di Star Fleet prima che si dissipassero. Le tempeste ioniche possono apparire all'improvviso ed essere particolarmente violente, possono causare minimi danni così come provocare danni ai sistemi primari o decessi a bordo. Possono verificarsi sia nei viaggi normali che nell'iperspazio. L'uso degli scudi difensivi può fornire qualche protezione, ma in caso di tempeste violente i danni si verificheranno comunque.

Il programma "Star Fleet I" è prodotto e distribuito in Italia dalla C.T.O. s.r.l., via Indipendenza 40, Bologna.

Chi ha incastrato Roger Rabbit?

Chi non ha visto o perlomeno sentito parlare del film più sensazionale dello scorso anno? Sto parlando, per chi non lo avesse ancora capito, del film-cartoon "Chi ha incastrato Roger Rabbit?" nel quale si ipotizza la convivenza quotidiana tra personaggi umani e personaggi animati ed i cui personaggi principali sono un coniglio (Roger Rabbit) ed uno scalcagnato investigatore privato (Eddie Valiant). Il film racconta la storia di questo coniglio, divo del cinema animato, che viene sospettato di aver ucciso, per gelosia, un certo Mervin, genio della Fabbrica degli Scherzi. Il povero coniglio, braccato dalle terribili Faine dell'ancor più temibile Giudice Doom, si rifugia da Eddie e gli chiede di aiutarlo a provare la sua innocenza. Da qui iniziano le rocambolesche vicissitudini attraverso le quali Roger ed Eddie si trovano a passare per sfuggire al giudice ed alla sua Salamoia (liquido terribile in grado di dissolvere i Fumetti facendoli sparire per sempre) e per tentare di identificare il colpevole ed il movente dell'omicidio.

Il lieto fine è d'obbligo: il colpevole è il giudice Doom (rivelatosi poi anche lui un Fumetto) che aveva ideato il diabolico progetto di dissolvere con la Salamoia tutti i Fumetti ed anche Cartunia. A tale scopo aveva ucciso il povero Marvin affinché non rendesse noto il suo testamento nel quale regalava ai Fumetti la città di Cartunia. Ultimo atto dell'impresa di Roger ed Eddie è proprio il ritrovamento di questo prezioso documento.

Come è stato reso noto dalla stampa, questo film è costato un'enormità e la causa principale di tale costo è da ricercarsi nella tecnica adottata per la creazione delle parti animate che sono state eseguite tutte a mano ignorando completamente l'uso del computer. Quindi quale migliore rivincita se non quella di costruire un gioco che abbia per protagonista questo simpaticissimo coniglio!!

Passiamo quindi senza soffermarci ulteriormente a parlare di questo videogame prodotto dalla Silent Service ed immesso sul mercato dalla Buena Vista Software.

Gli effetti sonori e le musiche sono di qualità eccellente, per quanto riguarda la rappresentazione grafica del game possiamo semplicemente affermare che è stupenda.

Il gioco inizia con una serie di schermate grafiche raffiguranti Baby Herman che invita Roger Rabbit a ricercare il testamento del signor Marvin; il gioco prosegue con Roger che lascia la Maroon Cartoons alla guida di Benny, il taxi animato, diretto alla volta dell'Ink & Paint Club in cui canta la bellissima

moglie Jessica, attenzione però alle terribili Faine che sono dirette allo stesso club e che devono assolutamente essere precedute. Altre difficoltà sono rappresentate dagli automezzi che circolano in entrambi i sensi e dalle macchie di Salamoia che occasionalmente appaiono sulla strada.

Una volta arrivati al club si passa al secondo quadro del gioco che vede impegnati un certo numero di camerieri - pinguini che passano in mezzo ai tavoli distribuendovi fogli di carta (tra i quali c'è il ricercatissimo testamento) e bicchieri di whisky.

Raccogliete quanti più fogli potete sperando di prendere anche quello giusto e fate ben attenzione a non toccare il bicchiere di whisky se volete evitare il tremendo effetto che gli alcolici hanno sul povero coniglio.

Questa è una delle fasi più difficili del gioco ma, se riuscirete a superarla, passerete alla schermata successiva dove vi troverete nuovamente in macchina diretti alla volta di... non voglio rovinarvi ulteriormente la sorpresa di scoprire di volta in volta le tappe attraverso le quali dimostrare la vostra abilità di videogiocatori, pertanto vi saluto augurandovi buon divertimento!!

Sword of Sodan

L'americana Discovery Software non ci aveva mai entusiasmato granchè con i suoi prodotti per Amiga; sia la conversione di "Arkanoid" che il gioco arcade - puzzle "Zoom" avevano lasciato critico e pubblico piuttosto tiepidi nel giudizio. Quando abbiamo ricevuto Sword of Sodan, già l'impatto con l'immagine del prodotto, e cioè confezione, manuale, ecc., ci ha subito bendisposti e incuriositi: abbiamo subito avuto la sensazione di avere tra le mani qualcosa di veramente speciale.

In effetti, tali aspettative non sono state deluse, Sword of Sodan è senza dubbio un prodotto unico nel suo genere. La complessa ed intensa storia introduttiva ci conduce in un'epoca indefinita, antica e immaginaria in cui le forze del male, rappresentate dal necromante Zoras, hanno avuto il sopravvento sull'intero territorio dei Regni del Nord.

Ma prima che il malefico usurpatore ne venisse a conoscenza, i due figli gemelli neonati del re furono portati di nascosto lontano dalle abitazioni regali.

Allevati con amore da un vecchio guerriero, Brodan e Shardan impararono presto a maneggiare la spada; quando conobbero l'intera storia del padre e le vicissitudini che ne causarono la triste fine, i due gemelli giurarono vendetta.

Ecco, quindi, che in Sword of Sodan possiamo scegliere di impersonare Brodan o Shardan: entrambi con un fisico mozzafiato ed una semplice armatura, affrontano ogni pericolo armati di sola spada. In aiuto alle forze del bene compariranno in modo casuale delle ampolle contenenti filtri magici con effetti diversi: a volte offrono vite in più, a volte aumentano la forza, altre offrono un potere magico che

rende invincibili per trenta secondi ed infine possono uccidere il nemico più vicino. Il gioco si sviluppa su più livelli, ognuno caratterizzato da scenari e personaggi differenti. Partendo dalle mura esterne ben guardate, il percorso si snoda attraverso paesaggi ostili per giungere al famigerato castello di Craggamoore, ove ci attendono le prove più dure ed i nemici più pericolosi.

Già, comunque, ai livelli precedenti non mancano avversari di ogni razza: giganti, legionari, ancient punks per non parlare dei terribili pseudo - iguana che, con la lunghissima coda terminante con un aculeo mortale, non perdonano mai. Più divertente invece la passeggiata attraverso il cimitero: qui gli zombies non ci rendono la vita facile, ma l'ambientazione particolare merita la visita!

Scherzi a parte, ogni settore che si attraversa riserva non pochi colpi di scena, che spesso colgono di sorpresa l'eroe o l'eroina di turno. Anche se la trama non dice nulla di nuovo (ma è ormai molto difficile inventare una story-line originale per un videogioco), Sword of Sodan ha comunque dei pregi che lo rendono molto appetibile. Dal punto di vista della realizzazione grafica, dobbiamo ammettere che Sword of Sodan è ben fatto: ogni scenario è ricco di dettagli ben definiti.

Anche i protagonisti sono disegnati con molta cura, così come le schermate di apertura e di intermezzo tra i livelli.

L'animazione è senza dubbio la caratteristica più rilevante di questo gioco: a momenti si è quasi in dubbio di trovarsi di fronte ad un ottimo cartone animato! In più, a rendere maggiormente realistica ogni sequenza, il commento sonoro gioca un ruolo importante: i rumori di fondo, lo stridere del metallo.

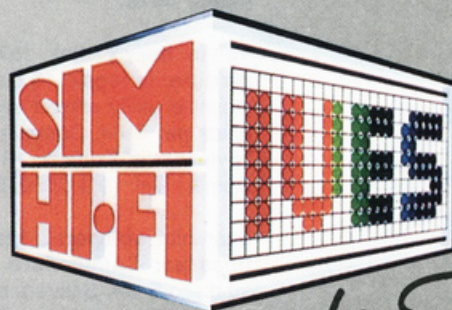
L'urlo di dolore dei feriti e gli effetti che accompagnano ogni movimento sono di grande efficacia.

Nell'insieme, il bello di Sword of Sodan è la precisione di ogni particolare; infatti, tra il vasto numero di giochi ormai disponibili su Amiga, è molto raro constatare tanta cura ed attenzione per ogni minimo dettaglio.

Come per ogni genere di prodotto, spesso sono le rifiniture a determinarne il livello qualitativo: Sword of Sodan è proprio uno di quei giochi che non riservano delusioni.

Ai giocatori più accaniti può forse sembrare piuttosto semplice, a confronto degli impossibili "spara e fuggi" dell'ultima generazione in effetti questo presenta minori difficoltà, ma comunque la vittoria finale per i più rimane un ambito trofeo!

Sword of Sodan (Discovery Software) è disponibile per Amiga a Lit.69.000 presso il servizio di vendita per corrispondenza SoftMail - via Napoleone, 16 - 22100 Como - tel. 031/300174.

MERCATINO

Cambio progammi selezionati per **C 64** e **Amiga**. Massima serietà. Scrivere o telefonare a Santì Mondo - Via Orsa Maggiore, 53 - 98057 Milazzo (ME) - Tel. (090) 9284863

Vendo **software** di qualunque genere **per Amiga** con le ultime novità in commercio. Massima serietà e risposta sicura. Richiedete gratis le liste. Vendo inoltre **joystick** compatibile. Vincenzo Viggiani - Via Isabella Morra, 10 - 85037 S. Arcangelo (PZ) - Tel. (0973) 811841

Vendo **C 128 + drive 1571** a L. 450.000; stampante Mps 802 grafica a L. 250.000; monitor Philips b/n a L. 100.000. Regalo insieme registratore, joystick, software e libri. Il tutto usato pochissimo. Angelo Turola - Via Comacchio, 38 - 44100 Ferrara - Tel. (0532) 60922

Compro **Amiga 500** per L. 400.000 oppure **C 128D** per L. 350.000. Richiesta valida per Genova città o Sardegna. Contattare Carlo Bazzoni - Via Budabest, 7 - 07100 Sassari - Tel. (079) 210314

Vendo **C 64**, registratore CSN, driver 1541, stampante 802, circa 2000 programmi e manuali. Inoltre walkman stereo Hitachi. Il tutto, in ottime condizioni, a L. 1.000.000. Claudio Borgonovi - Via Antonio Coppi, 6 - 00179 Roma - Tel. (06) 7828276

Vendo per **C 64 compilatore C** perfettamente funzionante e completo di istruzioni. Per informazioni scrivere, allegando bollo per risposta, al seguente recapito: Massimo Sisti - Via Mezzana, 64 - 27058 Voghera (PV) - Tel. (0383) 45767

Vendo **floppy disk 1570 Commodore**. In regalo 18 dischetti di giochi e utility oltre a una penna ottica per C 64. Contattare David Lovino - Via Erminio, 18 - 00174 Roma - Tel. (06) 765075

Scambio **software, idee, routines per Commodore 64**, solo su disco 5 1/4. Scrivere o telefonare ore pasti a: Marco Marinai - Via Costa, 49 - 56020 S. Maria a Monte (PISA) - Tel. (0587) 706593

Vendo **C 64 New** più registratore, joystick, guida riferimento, light pen, oltre 200 giochi. Tutto in perfette condizioni a L. 390.000. Rivolgersi a Diego Lorenzini - Via XX Settembre, 33 - 13011 Borgosesia (VC) - Tel. (0163) 22909

Cambio **game e utility per C64**, solo su disco. Inviare lista a: Giuseppe Di Lello - Corso Europa, 13 - 66054 Vasto (CH) - Tel. (0875) 60393

Vendo **C 128** completo di programmi, stampante, drive, registratore, joystick e porta dischi. Prezzo trattabile. Rivolgersi a Fiorenzo De Santis - Via S. Rosa, 18 - 03018 Paliano (FR) - Tel. (0779) 91473

Vendo o cambio **software per C 64**, disponibile su disco e cassetta. Ultime novità: Target Renegade, Impossible mission, eccetera. Telefonatemi: Fulvio Rossetti - Via Greve, 22 - 00146 Roma - Tel. (06) 5281545

Vendo **C 64** completo di drive, lightpen, monitor Philips e migliaia di giochi su nastro e disco e manuali. Tutto questo ad un prezzo d' amico. Telefonare ore pranzo a: Massimiliano Zanirato - Via Gallianico, 22 - 10146 Torino - Tel. (011) 7491347

Vendo **Amiga 1000**, sistema completo con imballaggi originali, 300 programmi e molti manuali, al migliore offerente. Telefonare ore pasti a Alessandro Serafini - Via Salvolini, 3 - 60129 Ancona - Tel. (071) 31273

Vendo **Commodore 128D con drive 1571** incorporato, monitor monocromatico, stampante Pd 80, joystick da 8, registratore 1516, programmi e manuali. Garanzia dicembre 88. Il tutto ancora imballato a L. 1.800.000. Telefonare ore pasti a: Massimiliano Aloisi - Via Modena, 21 - Cutrofiano (LE) - Tel. (0836) 662443

Garda **Amiga Team**, club sorto senza scopo di lucro, cerca scambi o contatti con altre associazioni o amighi. Scrivere a: Graziano Pasqua - Viale brescia, 2 - 25087 Salò (BS)

Vendo **Amiga 2000** più doppio drive, nuovo ancora in garanzia, a sole L. 2.000.000. Telefonare dopo le ore 19,30 a Piero Bertoldo - Via Lombardia, 1 - 25098 Verolanuova (BS) - Tel. (030) 931874

Compro monitor **1081 per Amiga**, preferibilmente modello vecchio. Cerco possessori di Amiga in zona Carpi per scambio di programmi. Enrico Filippi - Via F.LLI Saguatti, 4 - 41010 Carpi (MO) - Tel. (059) 660253

Cambio per **Amiga** qualunque tipo di programma. Per informazioni scrivere o telefonare a: Santi Mondo - Via Orsa Maggiore, 53 - 98057 Milazzo (ME) - Tel. (090) 9284863

Cambio **programmi per Amiga**. Inviare le proprie liste a: Roberto de Chaud - Via Sant'Elena, 198 - 16153 Genova-Sestri Ponente - Tel (010) 603726

Inviare questo coupon o una sua fotocopia a:
"Mercatino" AMIGA MAGAZINE
Gruppo Editoriale Jackson
via Rosellini, 12 - 20124 MILANO

____

____

____

Cognome ______________ Nome ______________

via ______________ n. ___ C.A.P. _______

Città ______________ tel. ______________

Firma ______________ Data ______________

AM6

CLASSIFICHE

DELUXE

di Maura Moncaro

Fra tante cose che ci capitano in redazione ci sono tanti dischetti senza etichetta contenenti programmi strani ,senza nome.

Il tutto poi senza un mittente.

Il tempo è poco per vederli tutti, ma a volte qualcuno ci incuriosisce.E proprio di uno degli ultimissimi che vogliamo parlarvi, anche se in maniera frettolosa e non approfondita dato che questo numero della rivista è in fase avanzata di impaginazione e qualcuno alle mie spalle (l'impaginatore) sta fumando nervosamente (strano, mi sembrava avesse smesso di fumare) aspettando quest'ultimo pezzo.

Alla "prima visione" del programma contenuto in questo dischetto mi è parso chiaro che si trattava di un programma di grafica, e stranamente mi è sembrato di riconoscerlo.

La prima schermata ci fa vedere le varie risoluzioni che si possono avere (320 x 200,640 x 200,320 x 400, 640 x 400); i numeri di colori che si possono ottenere (2-4-8-16-32-64......64???), boh! ; in più c'è un'opzione overscan. Dando o.k. si entra finalmente nel programma vero e proprio. Ma non sono finite le sorprese!

Overscan, tanto per intenderci, vuol dire proprio overscan cioè oltre i limiti fisici del video ; il che tradotto in spiccioli significa disegnare a tutto schermo e oltre, anche dove non si vede (utilissimo per l'uso video delle immagini create con Amiga).

Eppure mi sembra di conoscere questo programma! Il menù è fornitissimo(abbiamo una fretta del diavolo); tra l'altro un FLIP che ruota l'area completa dello schermo di 180 gradi sia in verticale che in orizzontale.

HANDLE ci permette invece di spostare un brush prendendolo in qualsiasi punto; poi

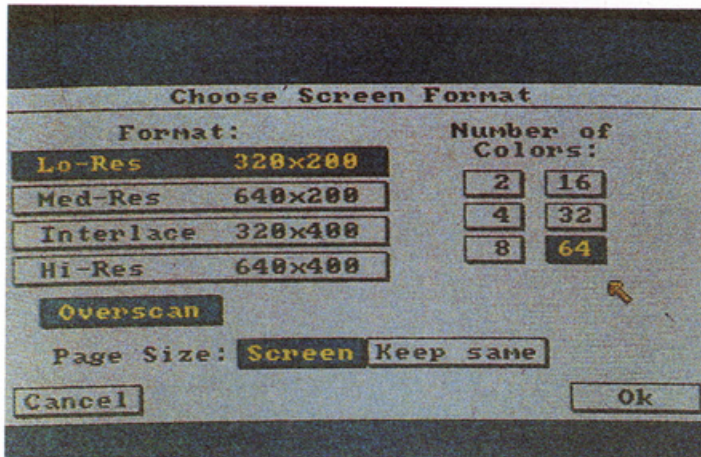
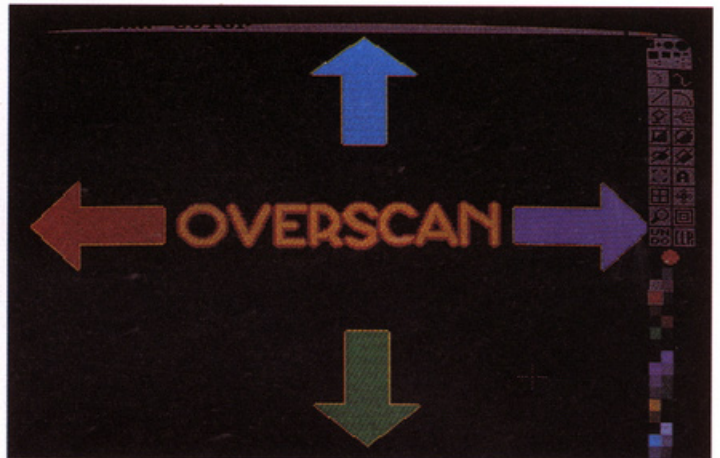
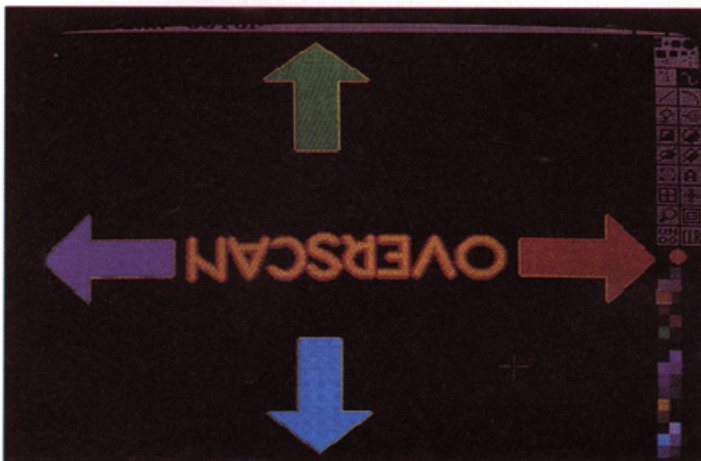
TINT ....boh?! .....

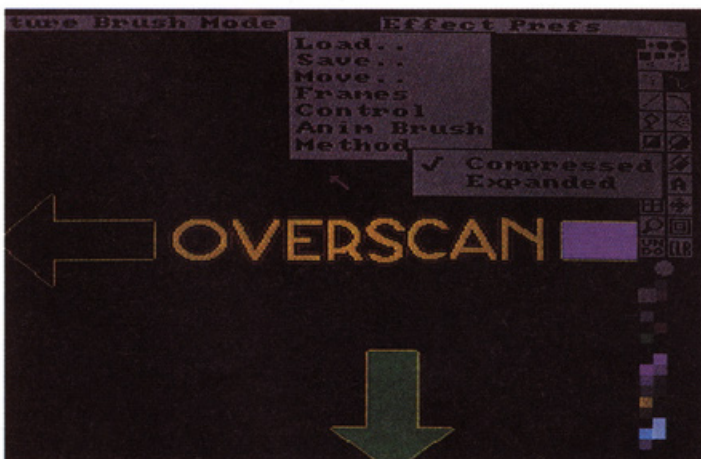
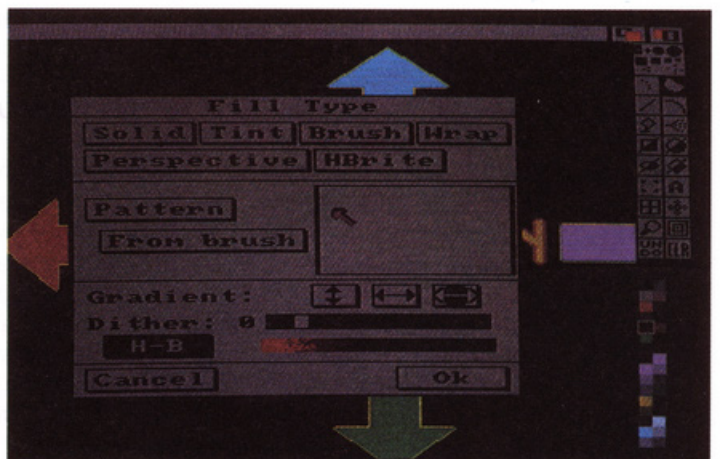
HBRITE.....

Animazione? Non mi sbaglio, è proprio un programma per creare animazione all'interno del programma stesso.

Le opzioni sono tante tra le quali la possibilità di creare uno spazio di qualsiasi formato con il fill automatico; il set di caratteri viene presentato in una maniera completamente nuova da altri programmi per Amiga già visti, e molto più completa e varia. E poi si possono usare 64 colori contemporaneamente in modo HALFBRITE.

Le opzioni sono tante e tutte da scoprire ma rimane poco tempo,(l'impaginatore!) eppure questo programma devo averlo già visto, e anche usato! NON SARA' MICA ... DELUXE PAINT III ???

Schermata iniziale

Overscan

Opzione FLIP 1

Opzione FLIP 2

Animazione

FILL

di Paolo Russo

Prosegue l'esperimento, iniziato nel numero precedente, di sdoppiamento del corso di AmigaBasic in due parti destinate la prima al neofita, la seconda all'aspirante esperto. Vedremo quindi, tanto per cominciare, come eseguire dei semplici calcoli, o piuttosto come scaricare tale incombenza sull'interprete Basic del nostro beneamato computer. Verranno anche illustrati i concetti basilari di variabile e di assegnazione. La sezione "avanzata" riprende invece l'argomento, appena scalfito nella scorsa puntata, della grafica amighiana.

Facciamo un po' di conti

Un'avvertenza prima di tutto: ogni tanto, nel corso degli esercizi pratici alla tastiera, potrà capitarvi di sbagliare qualcosa. Il Basic mostra allora una scritta nella parte alta dello schermo che indica la natura dell'errore ed alla cui destra compare un gadget con la scritta OK; per eliminare il messaggio e continuare a lavorare clickate sul suddetto gadget e attendete qualche secondo.

La puntata precedente ha illustrato quali differenze esistano tra l'output window e la list window, e di conseguenza tra i concetti di esecuzione e di memorizzazione di un ordine. Ricapitoliamo: un comando digitato nell'output window viene eseguito subito (e dimenticato), mentre si limita a venire memorizzato se scritto nella list window; in quest'ultimo caso la pressione di Amiga - R (o la digitazione della parola RUN nell'output window seguita dal tasto ENTER) provoca l'esecuzione dei comandi memorizzati.

Selezioniamo adesso la finestra di output, clickandoci dentro con il mouse co-

me di consueto, e chiediamoci cosa si può fare con il nostro computer. Inizialmente vedremo solo applicazioni piuttosto banali, semplicemente perché le cose utili richiedono troppe nozioni di base; procederemo quindi con molta calma. Se una persona non conosce per niente un determinato oggetto, conoscerà tuttavia almeno marginalmente oggetti simili, ed il modo più efficace per insegnare un concetto nuovo consiste appunto nell'appoggiarsi a concetti vecchi che gli rassomiglino almeno in parte, illustrandone nel contempo le differenze. Negli ultimi anni le calcolatrici hanno conosciuto una vastissima diffusione e noi ne approfitteremo.

Fingiamo adesso che il nostro Amiga sia una calcolatrice (sperando che non si offenda per un tale insulto) e vediamo come fargli fare dei banalissimi calcoli in Basic (senza quindi ricorrere al Calculator dello Workbench). Prima fase: quanto fa due più due? Un bel "print 2+2" senza virgolette e seguito da ENTER risolve questo titanico problema (d'ora in avanti sia l'assenza delle virgolette che la pressione del tasto ENTER o RETURN che dir si voglia saranno sottintesi). Piccolo inciso: il simbolo "+" appare sia nel tastierino di destra sia nella tastiera principale, ma in questo'ultimo caso per raggiungerlo dobbiamo premere contemporaneamente lo shift (esistono due tasti shift, il cui simbolo è una freccia verso l'alto, ai due lati della tastiera, che funzionano come il tasto delle maiuscole in una macchina per scrivere). Vediamo quindi che il comando print scrive sullo schermo il risultato della nostra operazione.

In realtà il print consente, più in generale, di stampare una qualunque espressione numerica, che può essere semplice come un singolo numero (come nel "print 123" della precedente puntata) o anche il risultato di una lunga sequenza di calcoli; possiamo rendercene conto con un "print 1+2+3-4". E per quanto riguarda prodotti e divisioni? Niente di più semplice, pensa il lettore cercando con lo sguardo i classici simboli del "per" e del "diviso" nella fornitissima tastiera Amiga. Niente da fare, non ci sono proprio. Nella quasi totalità dei linguaggi di programmazione esistenti, infatti, il "per" viene rappresentato dall'asterisco (*) ed il "diviso" dalla barra inclinata a destra (/), da non confondere con quella inclinata a sinistra, anch'essa presente. L'elevazione a potenza si ottiene invece con quello strano simbolo a forma di piccola V rovesciata che si trova solitamente sopra il 6 (almeno sulla tastiera anglosassone).

Compiamo un altro passettino. Sommiamo prima uno più due, moltiplicando poi il tutto per tre; digitiamo "print 1+2*3" e contempliamo la risposta fornita dal computer: 7. Qui qualcosa non va. Il fatto è che in matematica la moltiplicazione ha priorità sull'addizione e deve quindi essere eseguita per prima; il nostro intelligente interprete Basic adotta la medesima convenzione. Non crediate d'altra parte che ciò sia ovvio: altri linguaggi di programmazione eseguono rigidamente le operazioni da sinistra a destra, senza tenere in alcun conto le priorità. Anche per le calcolatrici vale lo stesso discorso; alcune rispettano le priorità e altre no. Per eseguire il nostro calcolo in Basic dobbiamo quindi digitare "print (1+2)*3". Le operazioni con la stessa priorità vengono eseguite sempre da sinistra a destra; * e /, in particolare, hanno la stessa priorità. Il numero di parentesi che si possono aprire le une dentro le altre è molto alto (eccedendo il limite dovrebbe apparire il messaggio "Formula too complex"). Il comando

print è in realtà tra i più complessi del linguaggio e lo stiamo sottoutilizzando; ne scopriremo le caratteristiche gradualmente. Per il momento è opportuno passare ad un altro argomento.

Memorie e variabili

Le calcolatrici, soprattutto quelle scientifiche, hanno di solito la capacità di immagazzinare uno o più numeri in entità chiamate memorie. È logico ritenere che anche il nostro ben più costoso Amiga possieda la stessa capacità. In effetti l'Amiga con i suoi 512K di RAM (Random Access Memory) possiede l'equivalente teorico di oltre 65000 memorie di calcolatrice, od un numero doppio se si riduce la precisione. Le memorie di una calcolatrice sono poche, numerate e scomode da usare, al punto che talvolta si preferisce farne a meno; in Basic si verifica l'opposto.

Le memorie del Basic e di quasi ogni altro linguaggio si chiamano variabili. Ogni variabile ha un nome che la contraddistingue e che può essere scelto dal programmatore con notevole libertà. Supponiamo di voler collocare il valore tre in una variabile di nome zip; digitiamo "zip=3" ed il gioco è fatto. Possiamo poi utilizzare il contenuto della nostra variabile come faremmo con un qualunque altro numero in successivi comandi, come "print 3*zip+2" o semplicemente "print zip". È sempre possibile cambiare il contenuto di una variabile in qualunque momento con una nuova assegnazione, ad esempio con "zip=-2", o "zip=3*4-1" o addirittura "zip=zip+1". Vediamo dunque di generalizzare. L'operazione di introduzione di un valore in una variabile si chiama tecnicamente assegnazione (assignment); il relativo comando si compone di un segno "=" con un nome di variabile alla sua sinistra ed una qualunque espressione numerica alla sua destra. Tale espressione, come nel caso del print, può contenere riferimenti ad altre variabili o perfino alla stessa variabile (come in "zip=zip+1"). Possiamo chiederci come si comporta il computer in un caso limite come l'ultimo menzionato. Ebbene, il computer calcola l'espressione sulla base del valore *attuale* della variabile e solo quando ha finito *tutti* i calcoli immagazzina il risultato finale nella variabile in questione.

Se quindi digitiamo "zip=4" e poi "zip=zip+1" e "print zip" vedremo un cinque sullo schermo. È importante sapere ciò per i seguenti motivi:

- il caso limite appena presentato non è affatto tale, nel senso che le sue applicazioni pratiche lo pongono all'interno della pratica programmatoria di ogni giorno, per quanto strano possa sembrare;
- l'esempio fornito illustra meglio di qualunque altro il dinamismo intrinseco nel concetto di variabile, in contrapposizione alla staticità tipica delle convenzioni matematiche, in base alle quali il valore associato ad un simbolo non cambia mai, ragion per cui x=x+1 può essere solo una relazione falsa o un'equazione priva di soluzioni.

Molta gente, infatti, tende a confondere le assegnazioni con le equazioni, che nei linguaggi di programmazione non compaiono. Il computer crea una variabile nel momento stesso in cui la si utilizza per la prima volta; zip non esisteva nella memoria del computer prima che noi digitassimo "zip=3". Il nome di una variabile è soggetto alle seguenti restrizioni:

- il primo carattere deve essere una lettera dell'alfabeto;
- tutti i caratteri successivi possono essere lettere o cifre;
- non sono quindi ammessi spazi vuoti nel nome;
- non viene fatta distinzione tra maiuscole e minuscole; zip, ZIP e Zip sono tre modi di scrivere lo stesso nome.

Alcuni possibili nomi di variabili sono i seguenti: a, x, salve, c1, zebra28fg4 e così via.

Un piccolo programma

Selezioniamo adesso la list window e digitiamo le seguenti linee:

```
a=3
b=4
c=sqr(a*a+b*b)
print c
```

Si può notare che le parole sqr e print sono state automaticamente riscritte dal computer in maiuscolo. Sqr, in particolare, è l'abbreviazione di square root; sqr(x) è quindi la radice quadrata di x. Il programmino appena scritto calcola l'ipotenusa c di un triangolo rettangolo con cateti uguali ad a e b. Premiamo Amiga destra - R ed il programma partirà scrivendo 5 all'interno dell'output window. Se vogliamo cambiare a e b (facendogli assumere ad esempio i valori 6 e 8) dobbiamo alterare il programma; per prima cosa bisogna far riapparire la finestra di list, nel caso che questa sia sparita come talvolta accade, premendo Amiga destra - L, dopodichè dobbiamo spostare il cursore nel punto desiderato. Il cursore, per chi non lo sapesse, è quella sbarretta verticale che si muove mentre scriviamo e che indica la posizione dove vengono inseriti nel testo i caratteri che stiamo digitando. Per spostare il cursore possiamo ricorrere al mouse, indicando con esso il punto desiderato e clickando.

Spostiamo dunque il cursore alla fine della prima riga e premiamo il tasto di cancellazione (quello con la freccia a sinistra che si trova subito sopra l'enter); il 3 di "a=3" dovrebbe essere sparito. Scriviamo un bel 6 al suo posto senza battere l'enter e premiamo il tasto con la freccia in basso; il cursore dovrebbe adesso trovarsi alla fine della seconda riga. I tasti con le quattro frecce fanno spostare il cursore e si chiamano, per l'appunto, tasti cursore (dovrebbe essere un fatto risaputo, ma non si sa mai). Cancelliamo il 4 sostituendolo con un 8 ed ecco cambiati i cateti. Se adesso premiamo la consueta coppia Amiga-R vedremo il prevedibile 10 sullo schermo. Tutto ciò è indubbiamente poco pratico. Bisogna forse modificare il programma ogni volta che i dati cambiano?

No, per fortuna. Esiste l'istruzione input. Sostituiamo "a=6" con "input a" e "b=8" con "input b"; selezioniamo poi l'output window, diamo un altro run con Amiga-R e vediamo cosa accade. Il computer stampa un punto interrogativo nell'output window ed aspetta che noi scriviamo qualcosa. Digitiamo "9" seguito da enter; ecco apparire un altro punto interrogativo. Scriviamo "12" con il solito enter ed ecco apparire un 15 sullo schermo. In effetti, se i cateti sono 9 e 12 l'ipotenusa è 15. Possiamo quindi dedurre che l'istruzione "input a" forza il computer a visualizzare un punto interrogativo, ad attendere che qualcuno scriva un numero, ad assegnare quindi tale numero alla variabile a ed a riprendere l'esecuzione delle rimanenti istruzioni. La stessa cosa accade con il secondo input, che svolge lo stesso compito per la

variabile b. L'istruzione input risulta quindi utilissima quando i dati su cui un programma opera devono cambiare ogni volta.

Per questa volta abbiamo finito. Se volete registrare permanentemente questo programmino potete usare l'opzione Save del menu Projects, come illustrato nell'ultima puntata. Riprendiamo adesso il discorso sulla grafica, rivolto principalmente ai programmatori semiesperti.

Sistemi di coordinate

Ogni computer possiede le sue convenzioni per quanto riguarda i sistemi di coordinate. Tutti i computer capaci di grafica ne possiedono almeno uno, che consente di individuare un punto rispetto ad un angolo dello schermo, ma le macchine che supportano le finestre ne prevedono almeno un secondo, relativo alla finestra in cui si sta disegnando. Ciò vale anche, in particolare, per l'Amiga. Parliamo innanzitutto del primo sistema di coordinate: l'origine si trova nell'angolo in alto a sinistra dello schermo e di conseguenza i valori delle ordinate aumentano verso il basso. Oltre a ciò, le coordinate sono sempre aderenti ai pixel reali, fatto questo indubbiamente positivo ma capace tuttavia di creare qualche problema quando gli stessi non sono quadrati. Lavorando ad esempio nella risoluzione di default (640x256) un rettangolo di dimensioni 100x100 non risulta affatto quadrato; bisogna allora raddoppiare la larghezza o dimezzare l'altezza.

Il primo sistema di riferimento trova impiego nel comando WINDOW per specificare posizione e dimensioni della finestra che si desidera aprire. In tutti gli altri casi, quando cioè si disegna realmente qualcosa, viene usato il secondo, che differisce dal primo solo per la posizione dell'origine, che si trova nell'angolo superiore sinistro della finestra e non dello schermo. Attenzione, però, che mentre il primo sistema di coordinate copre anche la barra superiore dello schermo in questione, il secondo non consente di tracciare sulla barra o i bordi di una finestra. Se quindi si apre uno schermo 320x200 ed al suo interno una finestra di pari dimensioni, i pixel indirizzabili all'interno della suddetta non sono affatto 320x200, bensì 303x187, dato che i rimanenti vengono mangiati dalla barra superiore e dai bordi, o 312x196 togliendo la barra superiore ed i gadget. Ciò risulta spesso seccante, ma si finisce con il farci l'abitudine.

I comandi grafici

I tre comandi grafici principali di ogni computer tracciano rispettivamente punti, linee e cerchi; la sintassi AmigaBasic di questi comandi somiglia molto a quella del Basica o GWBasic del mondo MS-DOS, data l'identità del loro creatore, la Microsoft:

```
PSET (x,y),[c] / PRESET (x,y),[c]
LINE (x1,y1)-(x2,y2),[c],[B[F]]
CIRCLE (x,y),r,[c],[s],[e],[a]
```

dove r=raggio, c=colore, s=start, e=end, a=aspect (rapporto tra gli assi). Le parentesi quadre indicano, come sempre, un parametro opzionale. PSET traccia un punto con il colore fornito o, per default, con quello corrente (che di solito è il numero uno, mentre lo zero è quello di fondo); PRESET fa la stessa cosa, ma usa per default il colore di fondo. LINE traccia una linea di estremi assegnati; specificando l'opzione B (box) si ottiene invece il rettangolo di cui quella linea rappresenta la diagonale, mentre specificando BF (filled box) si ottiene un rettangolo simile al precedente ma pieno anzichè vuoto (molto utile). CIRCLE traccia nel caso più semplice cerchi (o presunti tali; vedi oltre), nel più complesso archi di ellisse. Il raggio è inteso come il semiasse maggiore dell'ellisse; start ed end sono due angoli in radianti che esprimono l'inizio e la fine dell'arco di ellisse ed omettendoli si ha l'ellisse tutta intera, come ci si potrebbe aspettare. L'ultimo parametro (aspect) rappresenta il rapporto tra il semiasse inferiore e quello maggiore. Omettendo gli ultimi tre parametri si dovrebbe ottenere un cerchio.

In teoria, almeno. In bassa risoluzione (320x200 o 320x256) o in alta (640x400 o 640x512) i pixel sono, con ottima approssimazione, quadrati. Ciò accade solo in Europa e per pura coincidenza, dal momento che lo standard PAL comprime in verticale i pixel che sarebbero altrimenti troppo alti. Questo è il motivo per cui molte immagini sembrano schiacciate: sono state disegnate in NTSC e mostrate in PAL. Il nostro povero AmigaBasic ne risente alquanto; provate infatti a tracciare un cerchio in bassa risoluzione, ad esempio con un CIRCLE (200,100),50; la figura che ne risulta farebbe rivoltare Giotto nella tomba. L'AmigaBasic, infatti, non si accorge di essere in PAL ed applica spontaneamente il fattore correttivo 0.88 sull'asse verticale per compensare le deformazioni dell'NTSC.

Un
CIRCLE (200,100),50,,,,1
darà invece il risultato corretto, sia pur con un certo impiego di virgole per saltare i parametri intermedi. Analogo discorso vale per la media risoluzione tipo Workbench (640x200 o 640x256) per la quale un
CIRCLE (200,100),50,,,,.5
dovrebbe sortire l'effetto desiderato.

In tutti i comandi grafici appena visti si può inserire la parola chiave STEP subito prima di una coppia di coordinate; in tal modo esse vengono considerate relative a quelle dell'ultimo punto tracciato.

I limiti principali dei comandi grafici descritti consistono nella clamorosa assenza di una modalità di tracciamento in or esclusivo, assolutamente indispensabile in molti casi, e nell'aver definito raggio di un'ellisse il suo semiasse maggiore anzichè quello orizzontale o verticale; in tal modo, infatti, il tracciamento di una semplice sequenza di ellissi di eccentricità variabile ed indipendenti dal modo grafico diventa un'impresa allucinante, poichè ad un bel momento (che varia in base a diversi fattori) uno dei due assi supera l'altro ed i due si scambiano i ruoli.

Questo banalissimo problema, in passato, è costato al sottoscritto un paio d'ore del più infernale debugging, costellato di ipotesi, smentite immediate e controipotesi, di disperate correzioni empiriche dettate dal buon senso e di frenetici e deludenti test con i principali modi grafici; i vortici di ellissi risultanti sfidavano ogni tentativo di analisi del problema e l'intero arsenale di tecniche antibug sembrava del tutto inefficace.

Non avrei mai creduto che un problema così semplice potesse dare tanti guai; con ogni probabilità non sarebbe stato necessario tanto tempo se il manuale dell'AmigaBasic si fosse degnato di spiegare cosa intendeva esattamente per "raggio di un'ellisse", cosa che è invece apparsa evidente solo dopo molti tentativi.

EPYX INFOGRAMES

NASCITA DI UN NUMBER 1

Joan Brazier
Thomas Schmider
Gilbert Freeman
Bruno Bonnell

Bruno Bonnell e Gilbert Freeman

di Furio Lusnig

Nel 1984 la Epyx company (fondata da Jim Connelley nel 1979) ricevette più di 8 milioni di dollari per gli investimenti, il che le permise di sviluppare un nuovo mercato che riguardasse prevalentemente dei prodotti per home computer. Oggi questa compagnia è guidata da David S. Morse.

Morse è stato il fondatore di AMIGA COMPUTER dove vi rimase fino al 1975, è passato alla Epyx nel gennaio del 1987. Gilbert K. Freeman è l'addetto al controllo di tutte le operazioni finanziarie della compagnia. Meta dichiarata della compagnia è quella di raggiungere nel 1990 un volume d'affari di 100 milioni di dollari con un 20% di margine utile.

La Epyx Inc. è un'azienda leader nella creazione, sviluppo e marketing internazionale di prodotti di alta tecnologia per l'industria del tempo libero. Anche se la concorrenza nel campo degli home computer è altamente competitiva, la Epyx è riuscita ad imporsi, passando da un volume d'affari di 7 milioni di dollari nel 1983 a 27 milioni di dollari nel 1987.

I suoi prodotti sono venduti in Europa, Giappone, in Australia ed in Scandinavia. La compagnia inglese US GOLD di Birmingham ha venduto in esclusiva, negli ultimi tre anni, sul mercato europeo i prodotti della Epyx. Ricordiamo solamente alcuni dei successi: Winter Games, Impossible Mission e California Games.

Il successo di questa compagnia va attribuito ad una combinazione di fattori:

- alta qualità dei prodotti,
- una strategia di marketing innovativa,
- una costante diversificazione (accessori per computer come il joystick EPYX 500XJ; VCR GAMES come il Video Golf e Video California Games, ecc.),
- personale competente nei posti chiave della compagnia.

Gli studi della Epyx, di Redwood City in California, stanno preparando per la NEXT GENERATION dei nuovi prodotti ad alta tecnologia.

Il gruppo editoriale INFOGRAMES è nato in Francia nel giugno del 1983, creato da Bruno Bonnell e da Christophe Sapet. La sede di questa software house si trova a Lione in Francia e gran parte delle persone che vi lavorano si occupano della progettazione e dello sviluppo dei giochi. Nell'arco di soli cinque anni la compagnia si è proposta come guida nella creazione di software per il tempo libero, sviluppando e pubblicando successi

come LE CUBE, MANDRAGORE, LES PASSEGERS DU VENT, ecc.

Questo gruppo è il risultato della fusione di tre società leader nel settore dei video giochi per home computer:

- INFOGRAMES,
- ERE INTERNATIONAL,
- COBRA SOFT.

La ERE I. è la mente creativa del gruppo e si basa su una struttura che le permette di adattarsi rapidamente alle evoluzioni di un settore in continuo movimento. ERE I. è una specie di unità centrale che coordina una sessantina di autori indipendenti. Alcuni di loro collaborano da lungo tempo, ma sempre su idee personali le quali sono supervisionate in modo discreto dagli esperti di ERE I. che, all'occorrenza, forniscono le rifiniture.

L'ultima sezione è anche la più piccola ma questo non significa che sia meno vivace delle altre; gli adventure polizieschi della COBRA S. sono sempre ricchissimi di suspance.

Alcune cifre ci fanno comprendere il rapido sviluppo di questo gruppo. Nel 1983 il giro d'affari ha raggiunto il milione di franchi svizzeri, nel 1984 i 9 milioni, nel 1985 i 21 milioni, nel 1986 i 45 milioni e nel 1987 i 98 milioni. La crescita risulta annualmente del 300%. Durante lo stesso periodo il personale è aumentato da 4 a 5, poi da 30 a 50, e recentemente ha raggiunto le 110 persone.

Il gruppo è presente in 15 paesi (Australia, U.S.A., Giappone, Turchia, India, ecc.) ed ha realizzato oltre il 30% del suo fatturato con l'export nel 1987, mentre nel 1988 ha raggiunto il 60%.

"Il team INFOGRAMES è la ragione del nostro successo" afferma Bruno Bonnell. "La gente dice che una compagnia è una sola idea e un solo uomo; io vedo il successo come la combinazione di tre fattori: direzione, denaro e persone."

Sessanta persone operano in campi diversi come la programmazione, il marketing, le vendite, la pubblicità, la comunicazione, la grafica e la musica. Ma tutti insieme lavorano per raggiungere l'identica meta: produrre opere di alta qualità. La struttura editoriale ricopre tutti gli aspetti della pubblicazione e vi lavorano più di 40 persone. L'età media del personale è di 26 anni.

I prodotti della ditta sono siglati da un armadillo, animale che vive da millenni sempre pronto ad adattarsi senza perdere la sua identità. Scegliendo questo simbolo, la INFOGRAMES intende evidenziare la propria volontà di adattarsi continuamente ai nuovi bisogni del mercato. Comunicazione e lavoro d'equipe sono le chiavi principali della dinamica della casa e l'età media dei suoi effettivi, unita a una struttura flessibile, le sue forze motrici. Un'equipe marketing composta da dieci persone concepisce, elabora e produce i nuovi prodotti informatici grazie ai quali la INFORGRAMES è stata la prima software house a collegare l'informatica al fumetto, al "giallo" e allo sport.

Qual è la meta di INFOGRAMES?

È quella di fare della programmazione la decima arte dopo la TV e i Comics.

La collaborazione tra questi due gruppi, che hanno in totale 15 anni di esperienza, di creatività e di successi, è un chiaro segno del fatto che il mercato dei micro computer va visto sotto una prospettiva di carattere mondiale e non solamente di un paese o in scala puramente europea. L'INFOGRAMES ha realizzato questo concetto fin dal 1985 e con il passare degli anni ha incrementato il suo export fino a raggiungere il 60% nel 1988, utilizzando degli standard internazionali: PC, Apple IIGS, Macintosh, Atari, Amiga, ecc.

Questa determinazione ad essere presente sul mercato mondiale ha portato la INFOGRAMES a sviluppare degli approcci internazionali facendole guadagnare la reputazione di essere uno dei gruppi più innovativi, anche dal punto di vista tecnico, presenti sul mercato.

Il primo contatto tra INFOGRAMES e EPYX avvenne nel 1987, e l'incontro offrì l'opportunità ad entrambi i gruppi di apprezzare le reciproche conoscenze tecniche. I dirigenti di entrambi i gruppi rilevarono il fatto che le reciproche strategie offrivano molte similitudini, e dopo una serie di incontri si pose la questione di un'eventuale fusione tra i due gruppi. Nel giugno del 1988, durante il CES a Chicago, furono studiate le basi per un accordo tra le due compagnie. La fusione di queste due compagnie è la logica conclusione di una strategia d'approccio simile, tendente a considerare la tecnologia interattiva la principale area per lo svago del XXI secolo.

I risultati dell'accordo sono sia di natura strategica che finanziaria. Strategica, perché si è creata la più grossa struttura di publishing a statura internazionale con una diretta presenza sia in America che in Europa. Finanziaria, perché la fusione permette la costituzione di un gruppo leader nell'home computing dal punto di vista dei profitti e del volume d'affari.

Nel 1987 la EPYX deteneva il quarto posto nel mondo in termini di volume d'affari. Insieme, INFOGRAMES e EPYX detengono teoricamente il primo posto. La meta di questo nuovo gruppo è quella di divenire il NUMBER ONE nella pubblicazione dei giochi nel 1990. Anche se la principale attività sarà legata al software, il gruppo INFOGRAMES - EPYX ha già iniziato e continua la ricerca in nuovi settori di attività: video dischi interattivi, CD, CDROM, VCR GAMES, ecc.

Le conseguenze di questo accordo per la INFOGRAMES è che verrà facilitata la conquista del mercato americano e verrà consolidata la posizione di leader che detiene sul mercato francese. Sul mercato europeo attualmente la vendita di macchine attraversa un periodo di calma in attesa di una rapida crescita. Il mercato americano ha attraversato una simile situazione negli anni 1985-86, riprendendo nuova forza negli anni seguenti. Gli economisti prevedono un andamento simile in Europa negli anni 1989-90.

Il mercato europeo è diventato un obiettivo prioritario per le software house. Il loro interesse è rinforzato anche dalla prospettiva di un mercato unico a partire dal 1992. Il nuovo gruppo intensificherà i propri sforzi per incrementare le vendite anche in quello che è il terzo mercato mondiale per quanto riguarda il software e cioè il Giappone. Verranno cedute le licenze di una serie di giochi da svilupparsi in Giappone tramite la SEGA e la NINTENDO. Queste due compagnie sono riuscite ad adattarsi a questo mercato altamente competitivo e che non lascia penetrare al suo interno alcuna compagnia straniera. Per eventuali contatti stampa ci si rivolga a: EPYX: Noreen Lovoi - 600 Gallveston Drive - P.O. Box 8020 - Redwood City, CA 94063, U.S.A. TEL.: 415/366-0606; FAX: 415/369-2999 INFOGRAMES: Sabine Robert - 84, rue du 1er Mars 1943, 69628 Villeurbanne cedex, Francia, TEL.: 78.03.18.46; FAX: 78.03.18.40

SoftMail

Vendita per corrispondenza di programmi originali ed accessori per computers.

VIA NAPOLEONA 16 - 22100 COMO - TEL. (031) 30.01.74

® SoftMail è un marchio registrato da Lago snc

Presentiamo in questa pagina alcune tra le <u>ultime novità disponibili</u> dal catalogo SoftMail. Ecco alcune informazioni utili per utilizzare il servizio SoftMail: è possibile effettuare ordini telefonicamente SOLO se è già stata effettuata una spedizione a proprio nome ed è stata regolarmente ritirata. Dal secondo in poi <u>accettiamo ordini telefonici</u>. Chi desidera notizie sulla disponibilità ed i prezzi dei prodotti che non compaiono in questa lista può chiamare lo (031)30.01.74 dalle 14:30 alle 18:00. SoftMail può organizzare la <u>consegna anche tramite corriere</u>: interpellateci per maggiori informazioni. Oltre alle ultime novità qui esposte, SoftMail offre l'intero catalogo delle seguenti case: Aegis, Cinemaware, EA, Microprose, Rainbird, SSI, SSG, Sublogic.

ACCESSORI

Copritastiera A500	25.000
Final cartdrige III	110.000
Flicker master	29.000
Joy. Slik stick	16.500
Joy. SpeedKing	29.000
Joy. SpeedKing Autof.	33.000
Joy. Tac 2	29.000
Joy. Tac 5	39.000
MouseMat tappetino	22.500
Coprimouse	20.000
Portamouse	12.500
Portadischi 3" (30)	34.000
Portadischi 5" (40)	37.000
SlimLine (tastiera 64)	49.000
Voice messenger C64	60.000

LIBRI/HINTS & TIPS

Alternate city clue specificare 8/16 bit	18.000
Bard's tale I clue	22.500
Bard's tale II clue	25.000
Bard's tale III clue	25.000
Black cauldron clue	18.000
Deathlord clue	telef.
Dungeon master clue	25.000
Elite clue	18.000
Graphic adv.creator hint	7.500
Mars saga clue	telef.
Might & Magic clue	25.000
Pool of radiance cle	20.000
Quest for clues	39.000
Sentinel world clue	25.000
Starflight clue	25.000
Ultima V clue	22.500
Wasteland clue	16.500

AMIGA

Heroes lance hints disk	*telef.*
Bard's tale hints disk	*telef.*
Aegis draw 2000	telef.
Aegis images	69.000
Aegis impact	125.000
Aegis sonix	110.000
Aegis videotitiler 1.1	185.000
Alternate reality: the city	69.000
Animation multiplane	125.000
Armageddon man	49.000
Audiomaster	85.000
Audiomaster II	telef.
Baal	29.000
Barbarian II	telef.
Batman	29.000
Battlechess	49.000
Black cauldron	9.000
Blitzkrieg at Ardennes	59.000
California games	25.000
Carrier command	49.000
Cell animator	telef.
Chrono quest	65.000
Comic setter	139.000
Comic setter art disks	telef.
Corruption	49.000
Cosmic pirates	telef.
Crazy cars	29.000
Def con 5	59.000
Defender of the crown	59.000
Deluxe print II	99.000
DigiPaint	99.000
DigiView GOLD	telef.
Director	99.000
Disk drive esterno	299.000
Dos2Dos	75.000
Dragon's lair (1 MByte, 6 disks)	75.000
Driller TUTTO italiano	59.000
Dungeon master (1 MB)	59.000
Elite	45.000
Empire	49.000
F16 Falcon	59.000
Fantavision	125.000
Fire & forget	39.000
Fish	45.000
Flight simulator II	99.000
Scenery disks	telef.
Galileo 2.0	99.000
Garfield	49.000
Gauntlet II	25.000
Grabbit	49.000
Hellfire attack	39.000
International soccer	39.000
Italy 90 soccer	39.000
Incr.s.sphere	49.000
Jet & Japan bundle	110.000
Joan of Arc	25.000
King of Chicago	49.000
LED storm	25.000
Legend of the sword	45.000
Life cycles vol.1	39.000
Light!Camera!Action!	99.000
Lords of the Rising Sun	telef.
Major motion	39.000
Maxiplan plus	299.000
Menace	39.000
Mickey Mouse	25.000
Modeler 3D	129.000
Murder on Atlantic	59.000
Nigel Mansell grand prix	49.000
Obliterator	45.000
Offshore warrior	39.000
Outrun	25.000
P.O.W.	59.000
PacMania	25.000
Phantom fighter	45.000
Pioneer plague	59.000
President is missing	49.000
ProSound designer	79.000
ProSound w/hardware	199.000
Prowrite 2.0	179.000
Publisher plus	125.000
Reach for the stars	59.000
Rebel...Chickamauga	telef.
Return to Genesis	39.000
Rocket Ranger	59.000
Roger Rabbit	69.000
SEUCK	telef.
Sex vixens in space	55.000
Sculpt/Animate 4D	telef.
Sentinel	49.000
Sinbad	59.000
Skyfox II	59.000
Starfleet	59.000
Starglider II	49.000
Superman	39.000
Sword of sodan	69.000
The bard's tale II	49.000
The works!	249.000
Three Stooges	59.000
Tracker	49.000
TV Sports: Football	59.000
Ultima IV	49.000
Universal military sim.	45.000
Data disks 1 e 2	telef.
Victory road	39.000
Videoscape 3D 2.0	250.000
Designs disks	telef.
VIP professional	199.000
Virus	29.000
Virus infection protection	89.000
WB Extras	69.000
Whirligig	39.000
Willow	59.000
World Cl. Leaderboard	25.000
Zak McKracken	59.000
Zing keys	25.000
Zoetrope	199.000
Zoom	45.000
3 Demon	145.000

COMMODORE 64 CASS.

After burner	18.000
Barbarian II	18.000
Batman	18.000
Dragon ninja	15.000
Exploding fist+	15.000
G.I.hero	15.000
Italy 90 soccer	18.000
Last Ninja II	25.000
MicroSoccer	telef.
Robocop	18.000
R-type	18.000
Serve & volley	22.000
Shoot'em up constr. kit	25.000
Stunt bike simulator	5.000
Tiger road	15.000
Total eclipse (ital.)	15.000
4 soccer simulator	18.000

COMMODORE 64 DISCO

ADD: Heroes of the lance	telef.
ADD: Pool of radiance	59.000
American civil war III	49.000
Barbarian II	25.000
Deathlord	35.000
Driller	29.000
Echelon (con LipStick)	59.000
Eternal dagger	35.000
Fast break	29.000
Fish	29.000
Game over I+II	25.000
GeoPublisher	100.000
Grand Prix Circuit	telef.
Gulf strike	49.000
Home video producer	69.000
Italy 90 soccer	25.000
LED storm	15.000
Mars saga	35.000
McArhtu's war	49.000
MicroSoccer	telef.
Neuromancer	39.000
One on one II	29.000
Powerplay hockey	29.000
Red storm rising	49.000
Rocket Ranger	telef.
Shoot'em up constr. kit	35.000
Stealth mission	75.000
T.K.O.	29.000
The bard's tale III	35.000
Total eclipse (ital.)	20.000
Ultima V	49.000
Wasteland	39.000
Wec Les Mans	21.000
Who framed Roger Rabbit	telef.
Zak Mc Kracken	39.000
4 soccer simulator	25.000

COMMODORE 128 DISCO (128K, 80 COL.)

"C" Language	99.000
Basic 8.0	75.000
GEOS 128	telef.
Mach 128	125.000

OFFERTE SPECIALI

(FINO AD ESAURIMENTO)

		Valore	A sole
FIREBIRD SPECIAL			
AMIGA	Bubble Bobble+Whirligig+Enlightenment	107.000	75.000
C64 cass.	Savage+Intensity+Sentinel	48.000	33.000
C64 disco	Savage+Intensity+Sentinel	65.000	39.000
AEGIS ENTERTAINMENT			
AMIGA	Arazok's Tomb+Ports of Call	120.000	89.000

DEFENDER OF THE CROWN C64: cass. 12.000 disco 15.000

... SOFTMAIL TI DA' DI PIU'.

<u>*BUONO D'ORDINE*</u> da inviare a: **LAGO DIVISIONE SOFTMAIL**, VIA NAPOLEONA 16, 22100 COMO, FAX (031) 30.02.14, TEL (031) 30.01.74

Desidero ricevere i seguenti articoli:

☐ Addebitate l'importo sulla mia CARTASI/MASTERCARD/VISA/AMERICAN EXPRESS nr. ____________ scadenza ____________

☐ Pagherò al postino in contrassegno

Titolo del programma	Computer	Cassetta/disco/accessorio	Prezzo
		Contributo spese di spedizione Lit.	5.000
		TOTALE LIT.	

ORDINE MINIMO LIT. 20.000 (SPESE ESCLUSE) AMJ

Cognome e nome ____________

Indirizzo ____________ Nr. ____

CAP ______ Città ____________ Prov. ______ Telefono ____________

FIRMA (SE MINORENNE QUELLA DI UN GENITORE)

VERRANNO EVASI SOLO GLI ORDINI FIRMATI

AM6

di Paolo Russo

CORSO di ASSEMBLY

Sesta Parte: Il set di istruzioni

Nelle prime cinque puntate di questo corso abbiamo fatto la conoscenza del microprocessore MC68000 e delle sue principali caratteristiche. Non si puo' imparare una lingua straniera senza mandare a memoria lunghe liste di vocaboli; cio' è vero nche per l'Assembly, la cui comprensione è subordinata all'apprendimento di tutto (o quasi tutto) il suo set di istruzioni. Tuttavia, se fin dalla prima puntata il lettore fosse stato immerso nelle istruzioni senza neanche sapere cosa fosse un registro o un modo di indirizzamento, vi sarebbe probabilmente annegato. Non è pero' nemmeno possibile spiegare, ad esempio, cosa sia un indirizzo effettivo senza aver mai mostrato neppure un'istruzione; il classico serpente si morde la coda. Ecco per-ché fin dalle prime puntate sono state presentate al lettore alcune istruzioni del processore, con l'intenzione pero' di riservare alle stesse una trattazione piu' estesa e completa appena possibile. Il momento è arrivato.

I subset

Le istruzioni di un microprocessore possono essere raggruppate in diverse classi, che chiameremo subset, elencate qui di seguito:
- trasferimento dati;
- istruzioni aritmetiche;
- istruzioni logiche;
- shift e rotazioni;
- gestione di singoli bit;
- istruzioni di controllo;
- varie ed eventuali.

Il lettore ha già ricevuto una sommaria descrizione di qualche istruzione aritmetica e di trasferimento dati e di tutte quelle di controllo, ma un bel ripasso non dovrebbe nuocere a nessuno.

Il punto della situazione

Prima di addentrarci nella foresta dei subset riassumiamo brevemente le nozioni piu' importanti incontrate finora:
- il 68000 puo' operare su byte (8 bit), word (16 bit), long word (32 bit) e, in alcuni casi, su singoli bit;
- il tipo di dato su cui un'istruzione deve operare viene scritto subito dopo lo mnemonico dell'istruzione nella forma .B, .W o .L;
- word e long word esistono, in memoria, solo a indirizzi pari;
- il processore possiede 16 registri, metà dei quali orientati alla memorizzazione di dati generici (D0-D7), la restante metà a quella degli indirizzi (A0-A7);
- tutti i registri sono a 32 bit, possono quindi contenere indifferentemente byte, word

o long word di dati, ma i registri indirizzi non ammettono che gli si scriva dentro un semplice byte e convertono automaticamente le word in long word;

- quando si agisce su di un registro specificando .B o .W, l'operazione coinvolge sempre il byte o la word meno significativi del registro interessato;

- un operando (numero, registro o locazione di memoria) puo' essere specificato solo in una delle poche forme consentite, chiamate modi di indirizzamento (esempi: #3, D2, A4, (A2), (A7)+, 12(A5), -24(A0,D2.W), label(PC)...);

- ogni istruzione del processore puo' essere usata solo con una parte dei modi di indirizzamento esistenti. Le restrizioni imposte su di essi varia da istruzione a istruzione;

- la maggior parte delle istruzioni che manipolano dati funzionano unicamente su registri dati o locazioni di memoria;

- i registri indirizzi, dal canto loro, sono gli unici a poter essere usati come puntatori. Cio' deriva dal fatto che i modi di indirizzamento indiretti (quelli che consentono di accedere a locazioni di memoria il cui indirizzo è stato calcolato in qualche modo e posto in un registro) fanno riferimento quasi esclusivamente a tale classe di registri;

- il registro A7 coincide con lo stack pointer, puo' quindi essere identificato indifferentemente con le sigle A7 o SP, e NON deve essere alterato direttamente dal programmatore, a meno che quest'ultimo non sappia bene cio' che sta facendo;

- quando un'istruzione richiede due operandi, essi sono detti *sorgente* e *destinazione* e seguono il codice mnemonico dell'istruzione proprio in quest'ordine (MOVE.L D2,D3 muove una long word da D2 in D3 e non viceversa);

- non è possibile, per motivi tecnici, scrivere un dato in una locazione di memoria individuata rispetto al program counter (in altri termini, i modi di indirizzamento d16(PC) e d8(PC,i) sono vietati come operandi destinazione nelle istruzioni che alterano tale operando);

- esiste una serie di flag che memorizzano alcune proprietà del risultato ottenuto con l'ultima istruzione, e che vengono consultati dalle istruzioni di salto condizionato: Z (Zero), N (Negative), V (Overflow), C (Carry) e X (Extension);

- i programmi sono essenzialmente subroutine, che vengono richiamate dal sistema operativo, e devono quindi terminare con l'istruzione RTS;

- i programmi lanciati da CLI devono per convenzione porre nel registro D0, subito prima del RTS finale, uno zero se tutto è andato bene o, in caso contrario, un numero tanto piu' alto quanto piu' grave è l'errore verificatosi (5 = warning, 10 = error, 20 = fatal error...).

Passiamo adesso brevemente in rassegna i principali modi di indirizzamento:
1) Operandi non in memoria:
#n - l'operando è il numero n (immediato);
Dn - l'operando è il registro Dn (diretto);
An - l'operando è il registro An (diretto).
2) Operandi in memoria, l'indirizzo dei quali puo' essere specificato in uno dei seguenti modi:
n - l'indirizzo è n (assoluto);
(An) - l'indirizzo è contenuto in An (indiretto);
(An)+ - come (An), ma il registro viene poi incrementato di un valore pari alla lunghezza del dato (postincrementato);
-(An) - come (An), ma il registro viene prima decrementato di un valore pari alla lunghezza del dato (predecrementato);
d16(An) - l'indirizzo e' d16+An, dove d16 è un numero con segno a 16 bit;
d8(An,i) - l'indirizzo è d8+An+i, dove d8 è un numero con segno ad otto bit e i (indice) è un qualunque registro, seguito da .W o .L (indicizzato);
d16(PC) - l'indirizzo è d16+PC (program counter), ma spesso il numero d16 viene rappresentato da una label che marca un punto del programma (relativo al PC);
d8(PC,i) - l'indirizzo è d8+PC+i.
3) Operandi "strani", usati solo con alcune istruzioni;
SR - l'operando è lo status register (MOVE, AND, OR, EOR, supervisor mode only);
CCR - l'operando è il byte meno significativo dello SR, il Condition Code Register (MOVE, handle with care...);
USP - l'operando è lo stack pointer utente (MOVE, supervisor mode only);
offset - solitamente specificato con una label che marca un punto del programma (BRA, BSR, DBRA; sempre relativo al PC);
range - lista di registri (MOVEM).

Ribadiamo inoltre, ancora una volta, il concetto filosofico fondamentale dell'Assembly: il tipo di un dato non è associato al dato in sè, ma alle istruzioni usate per manipolarlo. Nè il processore, nè l'assembler protestano se diamo in pasto un numero privo di segno ad una istruzione concepita per i numeri dotati di segno, ma entrambi partono dal presupposto che il programmatore sappia sempre cio' che sta facendo; cio' consente l'esistenza di molte preziosissime scorciatoie ed altrettanti catastrofici bug. L'abilità di chi usa l'Assembly consiste nell'ottenere le prime senza i secondi. Nei primi tempi accade di regola il contrario, ma si tratta solo di una fase transitoria. È bene comunque notare che non è tanto importante essere bravi nello scrivere software evitando i bug (in Assembly è un'impresa impossibile, se non in casi particolarmente banali) quanto essere abili e tenaci nello snidarli. Dei tipi piu' diffusi di bug e delle tecniche per individuarli parleremo in futuro.

Il subset di trasferimento dati

L'istruzione fondamentale per trasferire dati da una parte all'altra del computer è MOVE, affiancata da un certo numero di sue forme secondarie che svolgono incarichi specifici. La sua sintassi è MOVE ea,ea (come spiegato in precedenza, ea significa effective address, che è un'infelice dizione per indicare la forma piu' generale che un operando puo' assumere. Gli operandi D2, #3, 4, (A5)+ sono alcuni esempi di effective address). Quest'istruzione è l'unica che consenta l'uso di un indirizzo effettivo per entrambi gli operandi. Ovviamente, la scrittura MOVE ea,ea deve essere considerata *cum grano salis* non sono chiaramente possibili sintassi come MOVE.W D2,#3 (un numero non puo' essere la destinazione di un trasferimento!), MOVE.B D0,A4 (non è lecito scrivere in un registro indirizzi un dato piu' corto di una word) o MOVE.L (A1),Tabella(PC) (è vietato scrivere in memoria con modi di indirizzamento relativi al PC).

L'istruzione MOVEQ #n,Dn pone un dato immediato (cioè un numero) ad otto bit in un registro dati. Essa si differenzia dalla piu' consueta MOVE.B in due aspetti:
- il numero viene automaticamente esteso a 32 bit e l'assegnazione coinvolge quindi l'intero registro, non solo il suo byte meno significativo;
- MOVEQ è corta (due byte tutto compreso) e quindi veloce; non per niente la Q sta per quick.

L'istruzione MOVEM (Move Multiple) trasferisce da o verso la memoria il contenuto di parecchi registri simultaneamente, ed è davvero utilissima quando è opportuno salvare sullo stack il contenuto di un buon numero di registri e successivamente recuperarli. Cio' accade soprattutto, rispettivamente, all'inizio e alla fine di una subroutine, allo scopo di limitare i "side effects" che si verificherebbero se le routine alterasse un registro impiegato nel programma principale. Non è pero' possibile decidere l'ordine in base al quale eseguire i trasferimenti, che segue quindi regole fisse. Ad esempio, MOVEM.L D2-D4/D6/A0-A3,-(A7) trasferisce nella parte piu' bassa dello stack il registro D2, seguito nell'ordine da D3, D4, D6, A0, A1, A2 e A3, che si trova piu' in alto di tutti. In parole povere, D0 sta sempre in basso, A7 in alto e tutti gli altri in mezzo. L'istruzione MOVEM.L A0/A1-A3/D2/D6/D3-D4,-(A7) avrebbe quindi avuto l'identico effetto della precedente. Le sintassi ammesse sono MOVEM range,ea - MOVEM ea,range. È possibile specificare sia .W che .L; nel primo caso, quando l'istruzione scrive nei registri su dimensione word, i valori letti dalla memoria vengono estesi a 32 bit prima di essere assegnati ai registri. Per qualche curioso motivo MOVEM non accetta il modo di indirizzamento -(An) come sorgente, nè (An)+ come destinazione.

L'istruzione MOVEP (che esiste unicamente nelle forme MOVEP d16(An),Dn - MOVEP Dn,d16(An), con .W o .L) legge o scrive in memoria a indirizzi alternati, un byte sì e uno no. Essa, come si puo' facilmente capire, non è di uso frequentissimo, essendo stata creata per risolvere situazioni hardware piuttosto particolari, che nell'Amiga non si verificano mai. Personalmente, credo d'averne fatto uso sull'Amiga un paio di volte (in KeyMacros, ad esempio).

MOVE e MOVEQ alterano i flag, se la destinazione non è un registro indirizzi. Cio' significa che per sapere, ad esempio, se un numero è zero, o se è negativo, basta trasferirlo in un registro. Il carry e l'overflow vengono sempre resettati.

L'istruzione CLR ea azzera l'operando indicato. Non ammette registri indirizzi (usare in tal caso SUB.L An,An) e non è consigliabile per cancellare l'intera long word di un registro dati per motivi di efficienza (meglio un MOVEQ #0,Dn). Agisce su byte, word o long word.

L'istruzione SWAP Dn scambia tra di loro le due metà di un registro dati (molto usata con le istruzioni di moltiplicazione e divisione), la EXG r1,r2 scambia tra di loro due registri qualunque (beh, lasciate in pace PC e SR), mentre il programmatore

medio scambia tra di loro EXG e SWAP con estrema facilità... Per fortuna l'assembler si accorge dell'errore dal numero degli operandi.

L'istruzione LEA ea,An (Load Effective Address) carica l'indirizzo dell'operando specificato (che deve quindi risiedere in memoria) in An, mentre PEA ea (Push Effective Address) lo deposita in cima allo stack. Ad esempio, LEA (A0),A1 e MOVE.L A0,A1 sono sinonimi, come anche accade per LEA 10,A2 e MOVE.W #10,A2 (beware of the "#"!), ma il LEA diviene molto utile con i modi di indirizzamento piu' complessi, consentendo al limite di eseguire in fretta una serie di somme che non hanno nulla a che fare con indirizzi. È particolarmente frequente la forma LEA d16(An),An, mentre il postincremento e il predecremento non sono ammessi (purtroppo). LEA e PEA non alterano i flag, come quasi sempre accade lavorando sugli indirizzi.

Potrà capitarvi di osservare in un listato strani mnemonici come MOVEI (Move Immediate) e MOVEA (Move Address), ma non fateci caso; trattasi di sinonimi di MOVE dovuti al fatto che, in linea di teoria, l'istruzione MOVE non ammetterebbe un dato immediato come sorgente o un registro indirizzi come destinazione; la Motorola ha creato istruzioni apposite per gestire queste situazioni. Naturalmente gli assembler convertono automaticamente i MOVE in MOVEA e MOVEI quando è necessario senza battere ciglio.

Il subset aritmetico

I microprocessori della vecchia generazione (quelli ad otto bit, tipo Z80 e 6502) consentono solo addizioni e sottrazioni. I micro a 16 bit, in genere, supportano anche moltiplicazioni e divisioni, ed il 68000 non poteva certamente fare eccezione.

L'istruzione ADD somma il primo operando al secondo, ponendo ovviamente il risultato nel secondo operando. L'istruzione puo' agire su byte, word o long word, ed ammette le seguenti forme: ADD #n,ea - ADD Dn,ea - ADD ea,Dn - ADD ea,An. Naturalmente tutti i flag vengono influenzati, compreso quello di estensione, che contiene il riporto generato dall'addizione. La forma veloce ADDQ #n,ea (anch'essa con .B, .W o .L) somma alla destinazione un numero compreso tra uno e otto; tale "strana" istruzione è in realtà la versione evoluta della INC di altri processori. L'istruzione ADDX Dn,Dn - ADDX -(An),-(An) (sempre con .B, .W o .L) somma alla destinazione sia la sorgente che il flag di estensione (inteso come zero o uno), che dovrebbe logicamente essere stato predisposto da una precedente addizione, allo scopo di realizzare operazioni in precisione multipla. In realtà 32 bit sono piu' che sufficienti nella maggior parte delle situazioni.

L'istruzione SUB è l'immagine speculare della ADD e tutto cio' che è stato detto per quest'ultima vale per la prima, compresa l'esistenza di SUBQ e SUBX.

Una forma particolare di sottrazione è realizzata dall'istruzione CMP (Compare, confronta) che sottrae il primo operando dal secondo senza alterare nient'altro che i flag; il risultato viene calcolato e gettato via. Anticamente per confrontare due valori si era soliti sottrarre l'uno dall'altro, ma così facendo si rovinava il secondo operando; CMP rimedia appunto a questo problema. Supponiamo di voler saltare alla label Zip se il contenuto di D0 (inteso come intero a 16 bit con segno) è compreso tra tre e sei:

```
CMP.W #3,D0
BLT.S 1$
CMP.W #6,D0
BLE.S Zip
```

CMP ammette solo le forme CMP ea,Dn - CMP #n,ea e non ha una versione veloce. Per quanto concerne le sigle dei codici di condizione usate nei salti, sarà opportuno mettere adeguatamente in luce che esse partono dal presupposto che sia il secondo operando ad essere confrontato con il primo; LT (Less Than) significa quindi "il secondo operando era inferiore al primo". Questo è uno dei punti piu' facili da dimenticare.

L'istruzione CMPM (An)+,(An)+ (Compare Memory) puo' essere usata per confrontare due zone di memoria (assai spesso stringhe di caratteri, nel qual caso si specifica il .B).

L'istruzione MULS ea,Dn moltiplica due interi a 16 bit con segno (Signed) generando un risultato a 32 bit; MULU assolve lo stesso compito per i numeri privi di segno (Unsigned). Il guaio è che ogni tanto capita di dover moltiplicare un signed per un unsigned (ma perché mai i progettisti non si fanno consigliare da programmatori esperti prima di progettare un set di istruzioni? Non sarebbe costato poi tanto creare una MULUS); cio' avviene sempre, di regola, nelle operazioni in precisione multipla, per esempio quando si moltiplicano interi con segno a 32 bit, cosa che richiede una subroutine. Questo è un capitolo molto importante dell'Assembly che passa solitamente sotto silenzio; ne riparleremo.

L'istruzione DIVS ea,Dn divide un long signed (32 bit con segno) contenuto in Dn per uno short signed (16 bit) generando un quoziente short signed nella word inferiore di Dn ed un resto short signed nella word superiore. Se ci interessa il resto possiamo quindi procurarcelo con un semplice SWAP. Se il resto non è nullo ha lo stesso segno del dividendo, contrariamente alle convenzioni dei matematici per i quali il resto non è mai negativo. Esiste naturalmente anche DIVU ea,Dn. Se il divisore supera il dividendo o il quoziente non entra in 16 bit il registro dati rimane inalterato ed il flag di overflow viene settato. E se il divisore è nullo? In tale sfortunata circostanza il microprocessore genera un exception, che è (piccola anticipazione) un interrupt di origine interna che provoca il passaggio al modo supervisore e la chiamata di una apposita subroutine che qualcuno deve avere predisposto all'uopo. Naturalmente, se tale routine non esiste (come quasi sempre accade), il malefico Guru sprofonderà nelle sue arcinote meditazioni. Morale della favola: prima di usare una divisione accertatevi sempre a priori dell'impossibilità che il divisore sia nullo. Per fortuna le istruzioni di divisione sono raramente necessarie.

Le istruzioni ADD, SUB e CMP con le relative forme veloci godono della ragguardevole proprietà di funzionare anche sui registri indirizzi.

Un'ultima nota sui tempi di esecuzione: somme e sottrazioni richiedono, se gli operandi si trovano già nei registri, solo 4T (ci sono sette milioni di T in un secondo) su byte e word, lo stesso tempo cioè che si impiega per trasferire un dato da un registro a un'altro, e 8T su long word; MULU e MULS richiedono circa 70T, DIVU e DIVS rispettivamente 140T e 158T. Non usate quindi queste istruzioni alla leggera.

AMIGA C

di Antonio Ferrillo

Dopo aver fatto conoscenza con gli schermi e le finestre ed aver scritto alcune funzioni generalizzate per il loro trattamento, questa volta lanostra libreria privata si arricchisce di alcune funzioni per la gestione del 'console.devicè e dei 'menus', i quali sono due importanti strumenti che il sistema mette a nostra disposizione per favorire il colloquio tra il programma in elaborazione e l'utente che opera alla Console.

Il 'console.devicè rappresenta il mezzo più potente e versatile per gestirel'I/O attraverso una finestra di Intuition, esso è un sottosistema dilivello più basso rispetto all'hardware che non Intuition stesso, ed oltread interagire con questi è al servizio anche di AmigaDos.

I 'menù, invece, sono quelle sequenze di scritte a forma di tendina che appaiono in alto sullo schermo quando viene premuto il pulsante destro delmouse, essi permettono all'utente di operare delle scelte, attivare dellefunzioni, di condizionare, insomma, dall'esterno l'esecuzione di unprogramma.

I devices

La parola 'devicè nel sistema Amiga è intesa in modo molto generico e sta ad indicare tutto quel complesso di fattori che concorrono a rendere possibile una operazione di trasferimento dati nei due sensi tra memoria centrale e qualsiasi dispositivo periferico.

Questi fattori comprendono : moduli elaborativi rom-residenti odisco-residenti, blocchi di controllo per gestire le richieste di I/O, i 'drivers' di device, cioè quei moduli che fanno da interfacciacon l'unità periferica, lo stesso dispositivo periferico e, naturalmente, l'attività elaborativa che viene implementata con l'aiuto dell'Exec.

Le comunicazioni tra il nostro Task e le routines di device avvengono utilizzando un blocco di controllo che assume il formato di una struttura ditipo 'IORequest' riportato in fig. 1, o, più spesso, di tipo 'IOStdReq' riportato in fig 2, che è una forma estesa della IORequest; ambedue si trovano definite nel file include <exec/io.h>.

Questa struttura, in realtà, non è altro che un messaggio la cui partevariabile assume un formato specifico tipico del device cui è indirizzato,infatti le interazioni tra il nostro Task ed il device che è stato attivatovengono regolate utilizzando il 'message passing' dell'Exec, per cui, prima di aprire il device dovremo creare il messaggio per inviare le richieste ed un 'portò di arrivo per riceverlo quando ci viene replicato aggiornato dal device.

Prima di essere impegnato con comandi specifici, un device deve esserea-perto utilizzando la funzione OpenDevice(), ed una volta terminate leoperazioni di I/O va chiuso utilizzando la CloseDevice().Le richieste vengono inviate utilizzando funzioni come la DoIO(), laSendIO(), la WaitIO(), la CheckIO() ed altre ancora, prima, però, avremoinserito nel messaggio IOStdReq il comando specifico che vogliamo far eseguire.

Usando la DoIO() noi facciamo sì che il controllo venga ritornato al nostroTask solo al completamento dell'operazione,con la SendIO(), invece, ilcontrollo viene ritornato immediatamente, prima cioè, che la richiesta vengacompletata, la CheckIO() serve per verificare il completamento di unaoperazione richiesta con la SendIO() e la WaitIO() mette il task in waitaspettando l'esaudimento di una SendIO().

I comandi da inserire nel blocco di controllo possono essere tipicamente :CMD_RESET che inizializza la periferica,CMD_READ che legge un datonumero di bytes, CMD_WRITE che trasferisce un dato numero di bytes,CMD_CLEAR che azzera un buffer di dati, CMD_STOP che mette in stato distop l'unità periferica, CMD_FLUSH che annulla le richieste di I/Opendenti.

Ogni device ha le sue caratteristiche peculiari anche in funzione del tipodi dispositivo interessato, per cui può richiedere un particolare settaggio del blocco di controllo, può supportare un proprio set di comandi e un proprio set di funzioni ad integrazione di quelli standard oltre arichiedere una specifica logica di gestione.

Il console.device

É il device che presiede all'immissione di dati via tastiera eall'emissione di dati su video via finestra AmigaDos o Intuition, è ingrado di convertire in formato ASCII i dati digitati e se li aspetta nellostesso formato per displayarli secondo le modalità suggeritegli inviando inscrittura delle particolari sequenze di caratteri di controllo.Esso può essere trattato come un qualsiasi file AmigaDos se si opera inquell'ambiente, o può essere gestito direttamente, in modo da sfruttarnetutta la potenza, in ambiente Intuition, ed è questo il caso che prenderemoin considerazione.

Prima di aprire il console device dobbiamo effettuare alcune operazionipreliminari : ovviamente avremo aperto una finestra cui associarlo,dobbiamo quindi creare un porto per ricevere le repliche dei messaggi di lettura o di scrittura se vogliamo leggere o scrivere, oppure due porti se vogliamo effettuare tutte e due le operazioni in maniera simultanea, similmente prepareremo uno o due blocchi IOStdReq dopo di che potremo chiamare la OpenDevice() per l'apertura.

Per creare un porto effettuiamo una chiamata alla funzione CreatePort()passandole come argomenti una stringa corrispondente ad un nome simbolicoche attribuiamo al porto e l'indicazione della priorità che è tipicamentequella del Task.

La CreatePort() allocherà memoria per una struttura di tipo MsgPort, lainizializzerà opportunamente e ci ritornerà il suo indirizzo che passeremo come argomento alla funzione CreateStdIO() la quale alloca memoria per unastruttura IOStdReq, la inizializza inserendo l'indirizzo della MsgPortricevuto come argomento nel campo appropriato della struttura Message cheè il primo membro della IOStdReq e finalmente ci ritorna l'indirizzo dellaIOStdReq stessa.

Ricordo per inciso che la CreatePort() e la CreateStdIO() fanno parte diquel gruppo di routines dette di 'supporto dell'Exec' che non sono presentiin 'exec.library', ma in libreria di compilatore e vengono inglobate nelprogramma in fase di linkage editor.

Naturalmente, se si vuol operare in lettura e scrittura simultanee, che è poi il caso più frequente, le chiamate a CreatePort() e CreateStdIO() sarannodoppie essendo doppie le strutture da inizializzare.A questo punto completeremo l'inizializzazione di uno dei blocchi IOStdReq con l'indirizzo della struttura Window, che abbiamo ricevuto dallaOpenWindow() quando abbiamo aperto la finestra, e la sua dimensione che inseriremo rispettivamente nei membri io_Data e io_Length e finalmente chiamiamo la OpenDevice() passandole come argomenti la stringa "console.device", che identifica il nostro device, il numero dell'unità (nel nostro caso = 0), l'indirizzo di una delle IOStdReq preparate prima, una sequenza di flags in una doppia voce (nel nostro caso = 0).

La Opendevice() ci ritorna un valore di 0 se l'apertura ha avuto successo odiverso da 0 se si è verificato un fallimento dovuto, nella maggioranzadei casi, ad un errore nella preparazione del blocco IOStdReq, essa altresìriempie in caso di successo i membri io_Device e io_Unit, i quali,se operiamo inlettura e scrittura, andranno copiati nei membri corrispondenti dell'altro blocco per far si che ambedue indirizzino le operazioni sullo stesso device.

Input dal console.device

Per leggere un dato numero numero di caratteri da tastiera occorre settare alcuni campi della IOStdReq; nel campo io_Command va il comando di lettura: CMD_READ, in io_Data va il pointer al nostro buffer di destinazione deicaratteri letti che avremo adeguatamente dimensionato, nel campo io_Lengthva il numero dei caratteri che vogliamo leggere.

Occorre tener presente che la pressione di un tasto può dar luogo allagenerazione anche di 5 caratteri, per cui con io_Length = a 5 avremobisogno di una sola lettura per accettare tutti i caratteri generati, conio_length = a 1 di letture ne dovremo affettuare 5, ad ogni modo ilconso-

le.device ci ritorna il numero dei caratteri letti nel campoio_Actual.

Preparato il nostro blocco IOStdReq di lettura lo passiamo come argomentoad una funzione di tipo DoIO() oppure SendIO(); se usiamo la DoIO() ilcontrollo al nostro Task viene ritornato solo dopo che l'utente si è decisoa premere un tasto facendo rimanere inutilmente attivo il Task stesso, percui è più opportuno utilizzare la SendIO() la quale avvia l'operazione e ciritorna subito il controllo cosicchÈ noi possiamo mettere il nostro Task instato di Wait (permettendo agli altri Tasks del sistema di girare piùvelocemente) in attesa che al nostro porto di ricezione delle repliche dei messaggi di lettura arrivi l'IOStdReq replicato che è segno di operazione avvenuta cioè l'utente, magari dopo lunga meditazione, ha premuto un tasto.

L'attività di trasmissione dati al Task da parte del console.device non silimita a questo : in seguito all'invio di particolari sequenze di controllocol comando CMD_WRITE egli può emulare l'IDCMPs di Intuition comunicandociinformazioni come la posizione del mouse,la selezione di un gadget o di unmenu, il cambio di una opzione delle Preferences, etc; egli, inoltre, puòtrasmetterci i caratteri della tastiera non convertiti in formato ASCII masecondo la codifica interna della macchina.

Output su console.device

Per inviare una stringa da displayare su video, od anche per inviare solouna sequenza di caratteri di controllo allo scopo di ottenere una data funzione occorre preparare la IOStdReq di scrittura : nel campo io_Commandinseriremo il comando CMD_WRITE, in io_Data il pointer alla nostra area discrittura, in io_Length il numero dei caratteri da scrivere, se inseriamo-1 vogliamo informare il device che il numero dei bytes è variabile e chela nostra area è null-terminata, nel caso della scrittura possiamo far ricorso alla DoIO() in quanto l'operazione è istantanea non essendoci tempi di attesa dipendenti dall'esterno.

Anteponendo alla nostra stringa da displayare una sequenza di caratteri dicontrollo possiamo determinare le modalità di display come la posizione delcursore, il colore del fondo e dei tratti del testo, lo stile dei caratteri.Queste sequenze di controllo sono introdotte da un particolare carattere cherisponde alla notazione esadecimale 0x9b, il quale per questo è chiamato CSI(control sequence introducer) e possono essere usate per ottenere numerose funzioni come lo scroll in alto o in basso,il clear del video o di una linea parziale o totale, muovere il cursore nelle quattro direzioni, lasparizione e l'apparizione del cursore e così via.

Per il dettaglio circa i caratteri di controllo vi rimando alla lettura del ROM KERNEL MANUAL così pure per le tabelle di codifica dei caratteri dellatastiera e per la gestione personalizzata delle mappe di tastiera.

Chiusura del console.device

Quando il console.device non ci serve più lo chiudiamo con unaCloseDevice() passandole come argomento il blocco IOStdReq utilizzato perl'apertura, liberiamo quindi la memoria allocata per la creazione dei porti e dei messaggi IOStdReq usando la DeletePort() col pointer alla struct MsgPort ottenuto in creazione dalla CreatePort() e la DeleteStdIO() col pointer alla struct IOStdReq ottenuto in creazione dalla CreateStdIO().

I menu

Sono il sistema più elegante e versatile a disposizione dell'utente percomandare dall'esterno l'esecuzione di un programma, essi vengono associatisempre ad una finestra anche se vengono visualizzati a partire dalla barraorizzontale dello schermo.

I menu si suddividono in tre livelli logici di selezione : le 'strips'rappresentano il primo livello, esse comprendono una catena di 'items' logicamente correlati fra di loro i quali rappresentano il secondo livello di selezione e vengono visualizzati in cascata sotto l'header della strip.

Ogni item può avere 'n' 'subitems' in subordine che rappresentano il terzolivello di selezione e che vengono visualizzati dall'alto in basso apartire dal lato destro dell'item cui sono sottoposti.

Tipologia degli item

Gli item e i subitem, dal punto di vista della filosofia di utilizzo sidividono in items o subitems 'azionè e items o subitems 'attributò.La selezione di un item azione si traduce subito in una azione da parte delprogramma e può avvenire ripetutamente, invece la selezione di un itemattributo si traduce nella determinazione di una condizione di elaborazionevalida dal momento della selezione e fino a che la selezione di un altroitem dello stesso tipo non ne determini un'altra attraverso il meccanismodella 'mutua esclusionè.

Mentre digito queste cose mi viene di pensare all'item 'Overstrikè che hoselezionato dal menù del mio editor che ha determinato la condizione discrittura sovrappostà e che rimarrà valida fino a che non selezioneròl'item 'Normal' che determinerà la condizione di 'scrittura con shift'.In pratica avviene che quando viene selezionato un item attributo essorimane selezionato e non può venire riselezionato fino a che non vieneselezionato un altro item che lo faccia diventare deselezionato (Perdonate loscioglilingua).

Questi particolari tipi di items, quando vengono selezionati, sono contraddistinti graficamente da un contrassegno di selezione simile ad una 'V' posto alla sinistra del testo.

Modalità di selezione

La selezione di un item avviene muovendosi tra gli items stessi colpuntatore del mouse tenendo premuto il pulsante destro e rilasciandolo quandoil puntatore stesso si trova sull'item che interessa.

É possibile determinare un modo di selezione alternativo usando lacontemporanea pressione del tasto 'right Amigà e di un altro tasto del cuivalore ASCII Intuition è stato informato nel modo opportuno al momentodella generazione dei menus.

É infine possibile disabilitare e abilitare un item alla selezione in funzione delle nostre esigenze.

Creazione di un menu

Occorre innanzitutto aprire una finestra mediante una chiamataalla routine di Intuition OpenWindow() senza dimenticare di settarel'IDCMP flag MENUPICK nella struttura NewWindow, naturalmente in precedenzaabbiamo aperto la 'intuition.library' mediante una OpenLibrary() per averaccesso, al momento dell'esecuzio-

ne, alle routines di Intuition residentiin ROM.

Settando il flag MENUPICK noi facciamo si che Intuition invii un messaggio alprogramma ogni volta che l'utente clicca sul pulsante destro del mouse per selezionare un menu.

Quindi bisogna fillare i campi di una serie di strutture di diverso tipo per informare Intuition su come strutturare le strips dei menus.Il primo tipo È rappresentato dalla struttura di tipo 'Menù,che definiscele strips di primo livello, ne occorre una per per ogni strip, essa È riportata in fig. 3 ripresa pari pari dal file include <intuition/intuition.h>.

Ricordo che i tipi di dato inusuali che si leggono sono delle ridefinizioni dei tipi standard del 'C' presenti nel file include <exec/types.h> che èopportuno inserire sempre in testa al programma prima degli altri files .h.Come il nome dei campi stessi ci suggerisce, settando questa struttura noi determiniamo la posizione relativa al margine sinistro dello schermo e le dimensioni del box-header della strip, vi inseriamo il testo da visualizzare,la leghiamo alla struttura successiva, se presente, mediante un pointer in modo tale da formare una catena di elementi collegati,infine,sempre mediante un pointer la colleghiamo al primo elemento della catena di items sottostanti.

La variabile Flag va posta a MENUENABLED per indicare che questa strip è normalmente abilitata alla selezione.

Le variabili Jazz e Beat ci rivelano le preferenze in fatto di musica degli autori di 'Intuition', a noi non interessano perchÈ sono per uso interno.Per definire gli items e i subitem occorre fillare la struttura di tipo 'MenuItem' riportata in fig. 4.

Qui inseriamo i dati relativi alla posizione relativa al box di testa e alle dimensioni, stabiliamo il legame con l'item o il subitem successivo (se c'È) e, se stiamo definendo un item, stabiliamo il legame col primo subitem dipendente.

Nel campo Command, se vogliamo, possiamo inserire un carattere ASCII corrispondente ad un tasto da utilizzare, insieme al tasto RIGHT AMIGA ed in alternativa al mouse, per la selezione dell'item, in questo caso nel campo Flags avremmo inserito un flag definito come COMMSEQ in <intuition/intuition.h>.

Il campo Flags può contenere numerosi altri flags in accordo con le nostreesigenze : se vogliamo che il nostro item o subitem sia di tipo attributoinseriremo CHECKIT e se vogliamo che risulti anche selezionato almomento della generazione del menu vi inseriamo CHECKED; se vogliamoche la rappresentazione grafica del nostro item sia un testo vi inseriamoil flag ITEMTEXT,se non inseriamo questo flag informiamo Intuition che lovogliamo rappresentare con una immagine.

Ancora stabiliamo le modalità di evidenziazione quando il puntatore passa sopra il nostro item : inserendo HIGHCOMP i colori del box vengonocomplementati, con HIGHBOX il box stesso viene riquadrato, HIGHIMAGEsignifica che viene visualizzata una immagine alternativa o un testoalternativo e perciò nel campo SelectFill va inserito un puntatore ad unastruttura di tipo 'IntuiText' per rendere un testo o di tipo 'Imagè perrendere una immagine.

Il campo MutualExclude serve per gestire il meccanismo della mutuaesclusione per quegli items che hanno il bit CHECKIT del campo Flags in on,in questo campo settiamo a 1 il bit che corrisponde al numero ordinaledell'item la cui selezione vogliamo escludere quando viene selezionato ilnostro item, questo numero è calcolato a partire da 0 nell'ambito della strip : se si tratta del primo item metteremo 1 nel bit 0, se si trattadel secondo item metteremo 1 nel bit 1 e così via.

Se abbiamo settato il bit ITEMTEXT,nel campo ItemFill inseriremo un puntatore ad una struttura di tipo IntuiText per la definizione del testo stesso, se non lo abbiamo settato vuol dire che vogliamo rendere una immagine, in questo caso dobbiamo inserire un puntatore ad una struttura di tipo Image.

Rimandiamo ad altra occasione la descrizione della struttura Image e analizziamo la struttura IntuiText di utilizzo senz'altro più frequente nel caso dei menus.

Essa è riportata in fig. 5 ed è generalmente utilizzata per descrivere lecaratteristiche dei testi da associare alle componenti di Intuition, perquesto motivo avremo occasione di incontrarla ancora.

Dopo essere stata fillata essa conterrà le informazioni circa il colore del fondo e dei tratti dei caratteri, del modo di display,della posizione della scritta all'interno del box,il pointer ad una struttura di tipo 'TextAttr' per la determinazione della Font di caratteri, il pointer alla stringa di caratteri da visualizzare e il pointer ad eventuali altre strutture IntuiTextper la definizione di ulteriori stringhe di caratteri.

Eseguito questo lavoro di filling possiamo finalmente effettuare la chiamata alla routine SetMenuStrip() passandole come argomenti il pointer ad una struttura di tipo Window che abbiamo ottenuto quando abbiamo aperto la finestra ed il pointer alla prima delle strutture Menu che abbiamo fillato.

Delete di una sequenza di menu

Una volta terminato il lavoro i menus vanno deletati prima della chiusuradella finestra, il lavoro da eseguire per sganciare una sequenza di menus infinitamente più semplice di quello necessario per la loro creazione : bastachiamare la ClearMenuStrip() col pointer alla Window.

Abilitazione e disabilitazione alla selezione

Per abilitare e disabilitare un item alla selezione Intuition mette anostra disposizione rispettivamente le funzioni OnMenu() e OffMenu() chepossiamo chiamare passando loro come argomenti il puntatore alla primadelle strutture Menu ed il cosiddetto 'numero di menù, che verràdescritto in dettaglio più avanti.

Intercettazione dei messaggi di selezione

Quando un utente preme il pulsante destro del mouse, Intuition ci invia unmessaggio di tipo MENUPICK e, nel campo Code della struttura di tipòIntuiMessagè (vedi fig 6), esso pone il numero di menu dell'item o subitemselezionato. La struttura IntuiMessage, che per adesso non descriveremo, è un messaggio così come la IOStdReq che abbiamo visto prima e di essa si serve Intuition per fornire al nostro Task quelle informazioni che abbiamo prenotato settando alcuni flags nel campo IDCMPFlags della struttura NewWindow quando abbiamo aperto la finestra.

Il campo Code della IntuiMessage è lungo 16 bits e, quando contiene ilnumero di menu, è così strutturato : i primi 5 bits

meno significativicontengono il numero ordinale della strip selezionata, i 6 bits centralicontengono il numero ordinale dell'item, gli ultimi 5 bits posti a sinistracontengono il numero ordinale del subitem se presente, oppure tutti 1(NOSUB) se l'item selezionato non ha subitems dipendenti.

Un valore di tutti 1 nel campo Code (MENUNULL) sta a significare chel'utente ha solo premuto il pulsante destro del mouse, ma non ha effettuatoselezione.

Per estrarre il numero ordinale del menu,dell'item e del subitem, vengono di solito usate delle macros definite in <intuition/intuition.h>, esse sono :MENUNUM,MENUITEM,SUBNUM che consiglio ai novizi del C di andare a guardare.

Le nostre funzioni

Per coloro che avessero avuto la sventura di aver mancato il numeroprecedente ricordo che lo scopo di questa serie di articoli è quello diguidare i lettori nella realizzazione di una libreria link-time di funzioniC per gestire Intuition, nel numero passato sono stati descritti glischermi e le finestre e sono state date le istruzioni per la catalogazionein libreria di compilatore sia per gli utenti Lattice che per gli utentiAztec.

Questa volta le funzioni che andiamo a descrivere sono raggruppate neisorgenti 'cons.c' e 'menu.c', presenti anche sul dischetto accluso insieme alsorgente e all'eseguibile del programma dimostrativo che le richiama edinsieme a due files include di cui parlerò più avanti.

cons.c

É una rielaborazione molto libera dell'esempio riportato sul RKM alla finedel capitolo dedicato al console.device,vi sono raggruppate numerosefunzioni di console.device ma non tutte,esse sono sopratutto orientate ademulare un terminale per applicazioni di tipo amministrativo-contabile conuso intenso di I/O su Console. Le funzioni di Read e di Write non offrono arrangiamento e formattazione dati limitandosi ad accettare e inviare stringhe null-terminate, ma questo può agevolmente essere ottenuto con le istruzioni e funzioni standard del C.

Non vi è bisogno di ulteriori commenti specie se avete letto con attenzionefino a questo punto, le funzioni stesse sono documentate all'internodel sorgente, solo una nota diretta ai nuovi lettori che riguarda anche lefunzioni menù : l'informazione relativa alla finestra cui associare il console.device e i menus viene trasmessa dal chiamante tramite un numeroche è poi quello d'ordine di apertura della finestra, questo numero vieneutilizzato dalle funzioni per ricavarsi l'indirizzo della finestrachiamando la routine Window_Addr contenuta nel modulo di gestione finestreresidente in libreria, coloro che non hanno questo modulo in libreriadevono effettuare le seguenti operazioni :

- generare una finestra nel chiamante
- passare come argomento alle funzioni il pointer alla finestra stessa ottenuto dalla routine di apertura
- modificare nelle funzioni chiamate la dichiarativa dell'argomento finestra da int a struct Window *
- sostituire, nel codice delle funzioni chiamate, la Call alla routine Window_Addr() col puntatore alla struct Window passato come argomento.

menu.c

Comprende le funzioni di generazione, rilascio, abilitazione edisabilitazione di un item.

Notate che se la finestra viene aperta su di uno schermo a bassarisoluzione il numero di strips visualizzabili si dimezza: purtroppol'esigenza di generalizzazione ha comportato un certo sacrificio in fattodi versatilità e qualche spreco di memoria.

Qualche parola sulla struttura del codice :
vengono inclusi gli header files necessari, sono dichiarate quindi le strutture di tipo IntuiText, MenuItem, Menu raggruppandole in arrays multidimensionali e sono inizializzate a valori di default con dei loops di 'for' annidati.

In questa fase vengono anche stabiliti i legami tra tutti gli elementi tramite i puntatori.

Segue la fase di filling in funzione dei parametri passati, anch'essarealizzata con dei loops di for annidati, in questa fase vengono spezzatii legami inserendo dei NULL nei puntatori agli elementi successivi o agli elementi dipendenti in modo da isolare quelli non richiesti.Il resto non ha bisogno di commenti.

demo.c

Il programma dimostrativo è strutturato in modo molto semplice, direi quasi didattico, le varie funzioni dimostrative sono temporizzate con lafunzione Delay() della 'dos.library' che non necessita di apertura allostesso modo della exec.library.

Gli errori vengono segnalati mediante un 'Alert' ed attivano la routine di fine programma che è utilizzata anche in caso di fine normale, in essa infatti viene chiuso solo ciò che è stato aperto durante l'esecuzione qualunque sia stato il suo esito.

Di particolare da segnalare è l'utilizzo dell''input devicè per emulare imovimenti del mouse.

L'input.device è quel device che monitorizza l'attività di altri tre : il 'gameport.devicè, il 'keyboard devicè e il 'timer.devicè fungendo da centrale di raccolta e smistamento delle informazioni che provengono dalle porte mouse/joystick,dalla tastiera e dal timer; esso è anche capace di generare un evento del tipo di quelli che gestisce, ed è questa proprietà cheè stata sfruttata nel programma.

Per finire una nota di ordine metodologico : come sapete, nel C le funzionisia interne che esterne che non siano di tipo 'int' vanno dichiarate, perliberarsi dall'incombenza di doverlo fare ogni volta che serve ed anche per procurarsi uno strumento di documentazione di veloce consultazione è opportuno raccogliere le dichiarative delle nostre funzioni di libreria in un file include '.h' da inserire in testa ai nostri programmi.

Ho preparato un file include .h contenente le dichiarative delle funzioniche abbiamo fino ad adesso collezionato che troverete sul dischetto colnome di 'prifunc.h' in due versioni : una per gli utenti Lattice ed una per gli utenti Aztec.

La versione per gli utenti Lattice contiene il dettaglio degli argomenti,questo perché i compilatori Lattice controllano il numero ed il tipo degli argomenti passati con la Call con quelli della dichiarativa generando un 'warning' in caso di discordanza.

Trasferite questo file sul vostro compilatore accanto al file <stdio.h>cambiandogli nome, se non vi piace quello che ha, e provvedete ad aggiornarlocon i futuri inserimenti di routine.

fig. 1

```
struct IORequest {
struct Message io_Message;
struct Device *io_Device;
struct Unit *io_Unit;
UWORD io_Command;
UBYTE io_Flags;
BYTE io_Error;
};
```

fig. 2

```
struct IOStdReq {
struct Message io_Message;
struct Device *io_Device;
struct Unit *io_Unit;
UWORD io_Command;
UBYTE io_Flags;
BYTE io_Error;
ULONG io_Actual;
ULONG io_Length;
APTR io_Data;
ULONG io_Offset;
};
```

fig. 3

```
struct Menu
{
struct Menu *NextMenu;
SHORT LeftEdge, TopEdge;
SHORT Width, Height;
USHORT Flags;
BYTE *MenuName;
struct MenuItem *FirstItem;
SHORT JazzX, JazzY, BeatX, BeatY;
};
```

fig. 4

```
struct MenuItem
{
struct MenuItem *NextItem;
SHORT LeftEdge, TopEdge;
SHORT Width, Height;
USHORT Flags;
LONG MutualExclude;
APTR ItemFill;
APTR SelectFill;
BYTE Command;
struct MenuItem *SubItem;
USHORT NextSelect;
};
```

fig. 5

```
struct IntuiText
{
UBYTE FrontPen, BackPen;
UBYTE DrawMode;
SHORT LeftEdge;
SHORT TopEdge;
struct TextAttr *ITextFont;
UBYTE *IText;
struct IntuiText *NextText;
};
```

fig. 6

```
struct IntuiMessage
{
struct Message ExecMessage;
ULONG Class;
USHORT Code;
USHORT Qualifier;
APTR IAddress;
SHORT MouseX, MouseY;
ULONG Seconds, Micros;
struct Window *IDCMPWindow;
struct IntuiMessage *SpecialLink;
};
```

di Luigi Manzo e Giovanni Michelon

Entriamo in questa seconda puntata nel vivo dell'argomento iniziando a presentare le strutture dati fondamentali messe a disposizione dal Modula-2; si tratta degli elementi fondamentali su cui operano le strutture algoritmiche e che, insieme a queste, costituiscono l'ossatura di un programma. Presenteremo inoltre un modulo di utilità che ci permetterà di semplificare al massimo le procedure di input/output, mettendoci nelle condizioni di scrivere i nostri primi programmini esemplificativi.

Identificatori e diagrammi di flusso

Come abbiamo visto nella parte introduttiva, l'utilizzo di una variabile in Modula-2 ne richiede la previa dichiarazione in un'apposita sezione del programma.

Sappiamo tutti che una variabile non è altro che una zona di memoria del calcolatore cui possiamo fare riferimento utilizzando un nome o, con termine tecnico, "identificatore". Un identificatore è una sequenza di caratteri; il primo deve essere una lettera, gli altri possono essere lettere (maiuscole o minuscole) o numeri; non vi possono essere simboli diversi da questi (%,!, spazi bianchi ecc.).

Un modo carino per esprimere quanto appena detto a parole è quello di usare un diagramma di flusso, rappresentato in figura 1.0. Il diagramma va letto da sinistra verso destra, seguendo le linee orientate: ad ogni biforcazione si ha un'alternativa possibile. Osserviamo che in questo caso è possibile ritornare indietro ripetendo un numero arbitrario di volte la sequenza di passi già eseguita. Ricorreremo più avanti a questo tipo di schemi per abbreviare la definizione di vari elementi del linguaggio.

Esempi di identificatori corretti sono:

```
Variabile1
TipoProva123
```

Identificatori scorretti sono:

```
Prova(4)
1x
Var intera
```

Dichiarazione di variabili

Ma come si fa in pratica a "dichiarare" una variabile? Esiste una apposita parola chiave, VAR, che serve a delimitare una zona del programma (la parte iniziale, di solito) in cui vanno specificati i nomi delle variabili ed il loro tipo. La sintassi esatta di tutto ciò è riportata nel diagramma di figura 1.1; qui ci limitiamo a riportare un esempio:

```
VAR
  Variabile1 : CARDINAL;
  Variabile2 : INTEGER;
  Var3, Var4 : REAL;
```

La regione di dichiarazione delle variabili termina con una qualunque altra parola chiave che inizia una diversa sezione del programma. Facciamo notare inoltre che l'ordine con cui si dispone il testo del programma è indifferente: è sufficiente rispettare gli elementi del grafo. L'unico criterio cui bisogna attenersi è quello della massima leggibilità e chiarezza dell'insieme; in pratica si finisce sempre con l'adottare una disposizione simile a quella adottata nell'esempietto visto. Bisogna però fare attenzione al "separatore" che deve esserci tra VAR ed il primo identificatore di variabile: un separatore non è altro che un carattere di spazio, o di ritorno a capo, o di fine linea; il grafo indica che ve ne può

essere più d'uno. Separatori inoltre possono essere inseriti ad ogni nodo del grafico; la loro presenza, sebbene vivamente consigliata per i soliti motivi di leggibilità, non è però indispensabile e per questo non è indicata nel grafo.

Osserviamo inoltre che il grafo prevede la possibilità di una lista vuota di dichiarazione: in pratica, un caso simile non ha alcuna utilità, ma è grammaticalmente corretto, ed il compilatore non segnalerà errore in questa situazione.

A questo punto non ci resta che analizzare il significato dei tipi cosiddetti "predefiniti", ossia che il compilatore, nello standard, rende immediatamente disponibili: avvertiamo subito che tutte le indicazioni che forniremo si riferiscono a macchine il cui bus dati è a 16 bit, come il nostro Amiga.

Tipo CARDINAL

Le variabili di questo tipo contengono numeri interi da 0 a 65535 (estremi inclusi), per cui vengono riservati dal compilatore 2 bytes.

E' pure disponibile un tipo LONGCARD che occupa 4 bytes e si estende da 0 a 4294967295. Bisogna prevedere nelle applicazioni che questo tipo di variabili non superino tali valori in fase di esecuzione del codice, pena una prematura uscita dal programma.

Tipo INTEGER

Si tratta di numeri interi, positivi e negativi, codificati in 2 bytes, appartenenti all'intervallo da -32768 a 32767. Anche qui abbiamo disponibile il tipo LONGINT che si estende da -2147483648 a 2147483647.

Vediamo le operazioni che è possibile eseguire sulle variabili di tipo INTEGER e CARDINAL; esse sono elencate nella tabella 2.0, con i simboli adottati.

Su somme, sottrazioni e moltiplicazioni non ci soffermiamo neanche. Notiamo invece che la divisione tra interi è indicata col simbolo DIV ed ha come risultato un valore intero; MOD invece restituisce il resto della divisione intera tra i suoi operandi, il secondo dei quali deve essere un CARDINAL. Entrambi sono da considerarsi operatori, esattamente come +, -, *; ad esempio, sono vere le seguenti uguaglianze:

```
13 DIV 5 = 2
19 DIV (-2) = -9
-17 MOD 3 = 1
```

Tutte queste operazioni restituisco un valore CARDINAL se entrambi gli operandi sono di tipo CARDINAL, mentre danno un tipo INTEGER se almeno un operando è di tipo INTEGER.

Tutto ciò è vero, in rispetto alle regole di tipizzazione, solo se l'operazione è effettuata tra due costanti; se compare anche una sola variabile, bisogna fare attenzione alla omogeneità degli operandi. Una deviazione da questa norma generale si ha se un operando è di tipo subranges (vedi più avanti):in tale caso però bisogna fare attenzione alla correttezza dei risultati, perché potrebbero venire effettuate indesiderate operazioni di complemento. Settando una opportuna opzione del compilatore però è possibile attuare un rigido controllo anche su questo tipo di variabili.

Considerazioni analoghe valgono per i rispettivi tipi LONG- : se almeno un operando è di tipo LONG-, anche il risultato sarà di tale tipo.

Consideriamo il seguente esempietto:

```
VAR
  c0,c1,c2 : CARDINAL;
  lc0,lc1  : LONGCARD;
  i        : INTEGER;
  li       : LONGINT;
BEGIN
   c0:= c1 DIV c2;
   lc0:= lc1 DIV c0;
   li:= lc1 MOD c0;
END nomeprogramma.
```

Non si tratta di un programma completo; gli mancano alcune parti fondamentali, e se anche venisse completato non darebbe molta soddisfazione all'utente perché è privo di qualunque forma di output; inoltre le variabili non sono inizializzate, per cui il loro valore viene pescato in una zona della memoria di lavoro del compilatore, e risulta pressochè casuale. Comunque le assegnazioni fatte risultano corrette grammaticamente: osserviamo alcuni fatti:

- come primo elemento abbiamo la sezione di dichiarazione delle variabili, aperta da VAR; consigliamo di seguire passo per passo la struttura dell'esempio, verificandola sul diagramma di fig. 1.1 al fine di prendere dimestichezza con questo tipo di rappresentazioni
- la fine di ogni istruzione completa è marcata da un ";"
- la parte algoritmica del programma, cioè quella che compie le operazioni sulle variabili (costituite da semplici assegnazioni nel nostro caso), è contenuta tra le due parole chiave BEGIN .. END; al posto di nomeprogr ci va il nome effettivo del programma che deve essere assegnato in precedenza
- l'assegnazione di una variabile avviene mediante il simbolo ":" seguito da un "=", ad indicare: "assegna alla variabile scritta a sinistra il valore dell'espressione scritta a destra. Il Basic per questa funzione utilizza semplicemente il simbolo "=", che invece in Modula-2 serve ad indicare l'identità logica di due espressioni.

Tipo BOOLEAN

Le variabili di questo tipo possono assumere 2 unici valori distinti: TRUE e FALSE; si tratta di due costanti predefinite che il compilatore riconosce senza bisogno di dichiarazioni precedenti. Questo tipo di variabili vengono codificate in un solo byte, e risultano utili ovunque vi sia da discriminare tra due situazioni alternative.

Le operazioni possibili su di esse sono riportate in figura 2.1: come è noto, l'operatore AND ha come risultato il valore vero (cioè TRUE) se e solo se entrambi gli operandi sono TRUE; OR invece è TRUE solo se almeno un operando è tale. NOT ha un solo operando, ed il risultato è TRUE solo se l'operando è FALSE, e viceversa.

Tipo CHAR

Le variabili di questo tipo sono (chi lo avrebbe mai detto, eh?) dei caratteri appartenenti allo standard ASCII; esempio:

```
VAR
  ch1, ch2, ch3, ch4:  CHAR;
BEGIN
   ch1:="g"; ch2:="G";
   ch3:='a'; ch4:='d';
END
```

Notare che il carattere può venire racchiuso indifferentemente tra apostrofi o

virgolette; ciò permette di definire delle stringhe di caratteri contenenti un apostrofo (e in tal caso la stringa sarà racchiusa da apici), oppure delle virgolette (la stringa sarà racchiusa da apostrofi). Ribadiamo che una variabile di tipo CHAR può contenere un solo carattere alla volta; vedremo prossimamente come è possibile definire dei tipi stringa.

Sulle variabili CHAR è possibile definire delle "operazioni" di confronto: il termine è, a rigore, improprio, perché il risultato di un confronto di questo tipo è un valore di tipo BOOLEAN, mentre un' operazione vera e propria ha come risultato un tipo comune agli operandi. In altre parole, in Modula-2 hanno perfettamente senso espressioni del tipo:
"a" < "z"

Ciò si comprende pensando al fatto che ad ogni carattere è associato il proprio codice ASCII; l'espressione riportata ha valore logico TRUE perché il codice del carattere "a" (97) è minore del codice del carattere "z" (122). Sono corrette ad esempio le seguenti assegnazioni:

```
flag1, flag2: BOOLEAN;
BEGIN
  flag1:= ("a" < "h");
  flag2:= ("B" = "d");
END nomeprog.
```

(La variabile flag2 ha valore FALSE).

In altre parole, il tipo CHAR risulta "ordinato"; vedremo più avanti il significato di questo termine.

Tipo REAL

Le variabili di questo tipo sono dette solo impropriamente reali, nel senso dell'analisi matematica: un numero reale infatti abbisogna, in generale, di un allineamento indefinito di cifre per poter essere rappresentato, mentre un calcolatore, come sappiamo, può memorizzare solo un numero limitato di cifre. Il tipo REAL, in realtà, indica un sottoinsieme finito di numeri razionali. Tali variabili vengono memorizzate in 4 bytes nel modo "floating point", e possono assumere valori compresi tra (3.40E-37, 3.40E+38), per positivi e negativi.

La struttura della sintassi di un numero REAL in Modula-2 è riportata in figura 1.2, e non sono necessari ulteriori commenti. Osserviamo che ogni numero reale deve contenere un punto decimale: ripetiamo, deve, anche quando non sembrerebbe necessario. Esempi di scritture corrette sono:
-1286.456
12.32 E37
12. E-7
3.

Anche in questo caso è disponibile un tipo LONGREAL, codificato in 8 bytes (sempre in floating point), il cui range è circa (3.40E-307, 3.40E+308).

Le operazioni effettuabili tra numeri reali sono le solite 4 aritmetiche: non vi sono commenti particolari. Anche qui, un'espressione può contenere CARDINAL, INTEGER e REAL insieme; se almeno un operando è REAL, il risultato è di tipo REAL.

Tipi enumerativi

E' possibile in Modula-2 dichiarare delle variabili che assumono soltanto certi valori prefissati dall'utente nella dichiarazione di variabile. Ci spieghiamo con un esempio:

```
VAR
  a: (Primavera, Estate, Autunno, Inverno);
```

La variabile a potrà assumere solo i valori elencati tra parentesi; saranno lecite assegnazioni del tipo:

a:=Primavera;
a:=Inverno;

Non sarà lecita l'assegnazione:

a:=Stagioni;

Facciamo notare che i valori Primavera, Estate, ecc. sono semplicemente degli identificatori, non delle stringhe di caratteri.

L'utilizzo di questo tipo di variabili ha lo

scopo di ottenere una maggiore chiarezza nei programmi, e permette al compilatore di controllare eventuali pericolose assegnazioni incrociate di variabili.

Tipi subranges

Si possono vedere come particolari categorie di tipi enumerativi; una variabile di questo tipo può assumere soltanto un sottoinsieme di valori contigui ed ordinati di un altro tipo definito in precedenza, chiamato tipo base. Facciamo subito un esempio:

```
VAR
  a: CARDINAL [21...11324];
```

La variabile a potrà assumere soltanto i valori interi da 21 a 11324 (estremi inclusi). La specificazione del tipo base (CARDINAL nel nostro caso), non è obbligatoria; lo diventa solo in caso di possibile ambiguità, come vedremo in seguito.

Facciamo notare che è implicito nella definizione data che il tipo base debba essere in qualche modo ordinato: ossia, che, preso un elemento qualunque al suo interno, debba essere chiaramente definito il suo successore. Tipi ordinati risultano essere i CARDINAL, gli INTEGER, i BOOLEAN (un po' particolari: vi sono solo due elementi, ed il successore di FALSE è TRUE !), i CHAR ed i tipi enumerativi, per i quali l'ordine è quello in cui sono elencati nella dichiarazione; ovviamente anche i tipi subranges saranno ordinati conseguentemente al tipo base.

Il tipo REAL non risulta, in base a questa definizione, un tipo ordinato; infatti, non è ben chiaro chi sia il successore, ad esempio, di 3.2343566 E13: esso dipende infatti dalla precisione di macchina e dal modo in cui le routines aritmetiche eseguono i troncamenti sulle cifre meno significative che escono dai fatidici 4 bytes dedicati ad un numero reale.

Dichiarazione di costanti

Può capitare, in un programma, di dover utilizzare per molte volte una certa quantità (un numero, o una stringa di caratteri), senza mai modificarla nel corso del programma stesso: in tali casi, si può ricorrere all'utilizzo di una "costante", ossia di un identificatore simbolico che può essere utilizzato al posto dell'espressione in tutto il programma, con evidenti vantaggi riguardo alla leggibilità e alla possibilità di ritoccarne il valore ovunque agendo su una sola dichiarazione.

La dichiarazione di una costante utilizza la parola chiave CONST, e si effettua di solito in testa al programma prima ancora della dichiarazione di variabile. Illustriamo la sintassi con un esempio:

```
CONST
  NumAvog = 6.023 E+23;
  Prompt = "Rispondi s oppure n";
  Molt = 10000;
```

Notate che l'assegnazione di una costante avviene con il simbolo "=" anzichè ":=", come per le variabili. Una costante può assumere un qualunque valore di tipo numerico (intero o reale), carattere o di stringa, senza nessuna precedente dichiarazione; in effetti, durante la compilazione viene effettuata una semplice sostituzione nel testo dell'identificatore con il valore ad esso corrispondente; si comprende pertanto che nel corso del programma non può venire effettuata alcuna modifica del valore della costante, pena un errore in compilazione.

Il compilatore inoltre non accetta costanti nel formato LONG-; cioè, se nella dichiarazione specificate un numero che esce dall'intervallo previsto per i CARDINAL (se la costante è intera), per gli INTEGER (se la costante è intera con segno), o per i REAL (se la costante ha punto decimale), allora avrete un errore in compilazione. Ad esempio sono errate le seguenti assegnazioni di costanti:

```
Err1 = 123456;
Err2 = -40000;
Err3 = 123.456 E 124;
```

Questa è una seccante limitazione del compilatore TDI.

Concludiamo indicando che, a differenza del Pascal, il Modula-2 permette di assegnare ad una costante il valore di un espressione. Ad esempio, è corretta la seguente dichiarazione:

2.0) Operazioni tra variabili intere

Operazione	Simbolo	Esempio
addizione	+	-4+7 = 3
sottrazione	-	12-21 = -9
moltiplicazione	*	21*(-5) = -60
divisione	DIV	17 DIV 3 = 5
modulo	MOD	23 MOD 9 = 5

2.1) Operazioni tra variabili logiche

Operazione	Simbolo
somma logica	OR
prodotto logico	AND
negazione	NOT

CostEsempio = 12*1243.34;

CostEsempio avrà valore 14920.08.

Un semplice esempio

Cerchiamo di riassumere quanto visto in questa puntata con un esempio operativo: si tratta di un programmino assolutamente idiota che ha il solo scopo di far provare al neofita l'ebbrezza della prima compilazione, e di fornire un quadro generale a chi voglia sperimentare le prime operazioni sui vari tipi esaminati. Non forniamo la copia del programma su disco proprio perché il suo scopo è quello di essere esaminato accuratamente, più che di essere semplicemente eseguito.

Osserviamo in testa al programma la dichiarazione:

MODULE Esempio1;

Essa definisce il nome del programma principale: tale nome va riportato (a differenza del Pascal) anche dopo la parola chiave END che chiude il programma.

In seguito è riportata la dichiarazione:

FROM MIO IMPORT ...

Essa serve a "importare" dal modulo MIO (acronimo per Modula Input/Output) la serie di funzioni elencate di seguito, necessarie per scrivere sullo schermo e leggere da tastiera i vari tipi di dati di cui si fa uso; il linguaggio infatti non ha disponibile immediatamente alcuna funzione di questo tipo. Ad esempio, la funzione wc (= write cardinal) serve a scrivere il valore di una variabile di tipo CARDINAL, e va usata specificando il numero di caratteri che si vuole dedicare al valore da scrivere, e ovviamente la variabile che si intende stampare. Vi sono numerosi esempi nel listato.

Il modulo MIO è un modulo di libreria costruito da noi, e serve a rendere il più semplici possibile le operazioni di I/O. Lo esamineremo a fondo nelle prossime puntate quando esamineremo le varie librerie standard; per adesso ci limitiamo a fornirvi tutto (file sorgente .mod e .def, e file compilati .sym e .lnk) su disco, in modo che i più esperti possano esaminarselo con comodo.

Continuando a scorrere il testo, troviamo la sezione di dichiarazione delle costanti, poi quella delle variabili, ed infine il programma vero e proprio, costituito da una serie di assegnazioni (eseguite con le funzioni di lettura da tastiera rc, ri, ecc.), da alcune operazioni sulle variabili, e dalla stampa dei risultati. Potete aggiungere o modificare alcune operazioni, in modo da rendervi conto dei risultati ottenibili; come al solito, la pratica è il miglior modo per apprendere costruttivamente. Vi sono numerosi commenti inseriti nel listato, inseriti tra i simboli "(*" e "*)"; leggeteli per comprendere esattamente il significato delle operazioni.

Passiamo ora alla fase di compilazione.

Interpreti e compilatori

Il Modula-2 è un linguaggio, come abbiamo già più volte ripetuto, compilato. Prima di dire come si fa a compilare un programma permetteteci di fare una rapida distinzione fra linguaggio compilato ed interpretato, in modo da stabilire la terminologia di uso corrente.

Ci riferiremo al nostro Modula-2 come campione dei linguaggi compilati e all'AmigaBasic per quelli interpretati.

Per entrambi i tipi di linguaggio il "programma" è costituito da un file testo, detto file sorgente, redatto con un editor. La distinzione avviene nelle fasi successive:

- nei linguggi tipo il Basic tale testo viene "letto" dall'interprete, che ne esegue le istruzioni passo a passo; come avrete senz'altro notato quando l'interprete non "capisce" qualcosa se ne esce con un tipico "syntax error";

- in Modula-2 invece il testo viene dapprima tradotto in un apposito formato e la traduzione è salvata in un file, detto file oggetto, prodotto dal compilatore; è in questa fase di traduzione che il compilatore segnalerà i vari "syntax error". Segue una seconda fase in cui il file oggetto viene completato di tutte quelle routines necessarie all'esecuzione del programma; artefice di questa fase è il Linker (collegatore, letteralmente), che produce il file eseguibile.

Vi sarà certamente saltato agli occhi che il secondo modo di procedere è certamente più noioso e lento, non solo, se il programma non funziona, dovrete riprendere tutta la trafila daccapo. Ma chi me l'ha fatto fare borbotterà qualcuno; beh, se pensate che tutto o quasi il software professionale (e anche quello ludico a volte) è prodotto con compilatori, vorrà evidentemente dire che "qualche" vantaggio c'è. Potremmo inoltrarci qui in una lunga disquisizione, ma lungi da noi far dell'accademia, preferiamo far risaltare i pregi e difetti di questo sistema man mano che affronteremo le caratteristiche del Modula-2.

Per il momento il nostro bilancio può essere il seguente:

- maggiore "noia" per la messa a punto del programma (0-1);
- il programma eseguibile è indipendente dall'interprete, quindi è possibile creare, per esempio, dei comandi CLI (1-1);
- minore occupazione della memoria e quindi migliore sfruttamento delle risorse hardware, infatti il programma compilato occupa solo lo spazio dato dalla sua lunghezza e quello riservato alle variabili, mentre un linguggio interpretato richiede anche la contemporanea presenza dell'interprete (osservate un po' quanto e "lungo" AmigaBasic) (2-1);

2 a 1 su campo neutro: un bel risultato; dobbiamo però far notare che il primo goal degli avversari è dovuto alla miopia dell'arbitro che non ha visto il fuorigioco!

Scherzi e punteggi a parte, non prendete per oro colato ciò che abbiamo appena detto: nostro intento infatti è quello di esaltare pregi e difetti, non di decidere per voi ciò che è meglio per voi (come invece più di qualcuno vorrebbe fare).

Una Ferrari Testarossa è indubbiamente superiore ad una 500, si rimorchiano più ragazze, è più veloce, ecc., se però dovete girare in città allora potreste avere degli inconvenienti, la macchina tende ad ingolfarsi, una banale amaccatura può diventare una grande seccatura... Il modo per fare la scelta migliore è quello di ampiliare al massimo le nostre conoscenze: ecco perché su questa rivista, come su tutte quelle che si rispettino, si cerca di dare il massimo spazio possibile ai vari linguaggi e alla cultura informatica in generale, piuttosto che alla cultura da videogiochi (che nel nostro caso tenderebbe a ridurre Amiga ad un gioco da bar, invece che premiarne le innovatività).

Come compilare

Bando alle chiacchiere! Supponendo

che abbiate preparato un dischetto per il Modula-2 come quello che vi abbiamo suggerito nel numero precedente, procedete come segue: scrivete con un editor il testo del programma. Salverete il file sorgente con il nome Esempio1.mod (.mod indica file sorgente) nel RAM: e sul vostro dischetto. Copiate in RAM: i files MIO.sym e MIO.lnk, inserite nel drive DF0: il dischetto con le librerie e impartite il comando:

ASSIGN M2: DF0:M2

Prima fase compilazione; comandate:

MODULA Esempio1.mod l

se tutto va bene trovate in RAM: i files Esempio1.lnk ed Esempio1.lst, il primo è il file oggetto, il secondo è il file sorgente con le linee numerate ecc. Questo file è generato dal compilatore per via di quella 'l' (sta per "list"); se avete commesso degli errori questi saranno segnalati in un apposito file Esempio1.erm, ma anche nel file Esempio1.lst, leggendo Esempio1.lst (con Ed o altri) troverete così gli errori commessi.

Per esperienza personale gli errori più comuni dei neofiti del linguaggio, ma anche dei veterani, sono: dimenticare un ';' o scrivere "a=b" invece di "a:=b".

Se non ci sono errori allora:

Seconda fase linkaggio; impartite:

LINK Esempio1 o

viene prodotto il file Esempio dal file Esempio.lnk, se avete fatto ciò che vi abbiamo detto, allora il programma si esegue chiamandolo per nome: Esempio1 (come per un comando CLI).

La 'o' comanda al linker di ottimizzare il codice, per ora basti sapere che viene prodotto un file eseguibile più compatto.

Qui dovrebbe filare tutto liscio come l'olio, se qualcosa non va allora significa che vi siete dimenticati di copiare MIO.#? nel RAM:.

AVVERTENZA:
se avete Amiga500 prendete la sana abitudine di stabilire la data, altrimenti il compilatore potrebbe rifiutarsi di lavorare! (Abbiate un po' di pazienza, ma non si può dire tutto subito!)

```
(*--------------------------------------------------------------------*)
MODULE Esempio1;                                          (* Ver 1.0 *)
(*--------------------------------------------------------------------*)
(* Questo e' un commento: tutto cio che viene scritto tra le parentesi *)
(* con l'asterisco viene ignorato dal compilatore                      *)
(*--------------------------------------------------------------------*)
FROM MIO IMPORT  rc, ri, rlc, rr, rlr, rs, wb, w, ws, wi, wc, wi,
                 wlc, wli, wr, wlr;

CONST
   pi= 3.14159;

VAR
   int1, int2, int3        : INTEGER;
   card1,card2,card3       : LONGCARD;
   r1                      : REAL;
   b1,b2                   : BOOLEAN;
   raggio, area, pilong    : LONGREAL;
   mese1, mese2            : (Gen, Feb, Mar, Apr, Mag, Giu,
                               Lug, Ago, Sett, Ott, Nov, Dic);
BEGIN
   (* Divisione tra interi--------------------------------------------*)
   ws (' Scrivi due INTEGER '); (* ws significa Write String   *)
   wl;                          (* wl porta il cursore a capo  *)
   ri (int1);
   ri (int2);
   int3 := int1 DIV int2;
   ws ('a DIV b = ');
   wi (int3,8);
   wl;

   (* Operazione di modulo tra LONGCARD ------------------------------*)
   ws (' Scrivi due LONGCARD '); wl;
   rlc (card1);
   rlc (card2);

   card3:= card1 MOD card2;

   ws ('a MOD b = ');
   wlc (card3,32); wl;

   (* Operazione di confronto tra reali -----------------------------*)
   ws (' Inserisci pi greco con 5 decimali esatti ');
   wl;
   rr (r1);
   b1:=  (pi=r1);     (* b1 sara TRUE se effettivamente pi e'    *)
                      (*   uguale a r1: FALSE in caso contrario *)
   wb (b1,6); wl;

   (* Calcolo in aritmetica reale lunga -----------------------------*)
   pilong := 3.14159265358; (* N.B: non e' possibile dichiarare*)
                            (* una costante cosi' lunga        *)
   ws ('Dammi il raggio '); wl;
   rlr (raggio);
   area:= pilong * raggio * raggio;
   ws("l'area del cerchio vale ");
   wlr (area,30); wl;

   (* Operazione di confronto su un tipo enumerativo ----------*)
   ws("Confronto fra variabili enumerative"); wl;
   mese1:= Gen;
   mese2:= Mar;
   b1 := (mese1 < mese2);
   wb (b1,6); wl;
   b2:= (mese1 > mese2);
   wb (b2,6); wl;
END Esempio1.
```

SCOPRILO FINO ALL'ULTIMO BIT

Scopri tutto quello che può darti il tuo computer AMIGA.
Ogni mese, dal GRUPPO EDITORIALE JACKSON, tre riviste che ti danno di più: AMIGA TRANSACTOR per i programmatori più smaliziati, AMIGA MAGAZINE per chi vuole conoscere il proprio computer sempre di più e AMIGA MAGAZINE GAMES per i giocatori più accaniti.
AMIGA TRANSACTOR, AMIGA MAGAZINE e AMIGA MAGAZINE GAMES: un panorama completo ed esauriente dell'universo di AMIGA, tre riviste per usare il tuo computer fino all'ultimo bit.

Scegli il meglio: scegli Jackson

PER I TUOI ACQUISTI RIVOLGITI AI PUNTI VENDITA UFFICIALI COMMODORE

LOMBARDIA

MILANO ALPHA COMPUTER Via Tavazzano 14 • AL RISPARMIO V.le Monza 204 • BCS Via Mantegani 11 • BRAHA ALBERTO Via P. Capponi, 5 • E.D.S. C.so Porta Ticinese, 4 • E.S.C. Via Roggia Scagna, 7 • FAREF Via A. Volta, 21 • FLOPPERIA SRL V.le Montenero, 31 • GIGLIONI LAURA Via D'Ovidio, 8 • GIGLIONI V.le Luigi Sturzo, 45 • LOGITEK Via Golgi, 60 • MARCUCCI Via Fratelli Bronzetti, 37 • MELCHIONI Via P. Colletta, 37 • MESSAGGERIE MUSICALI Galleria Del Corso, 2 • NEWEL Via Mac Mahon, 75 • RIVOLA Via Vitruvio, 43 • **S.ANGELO LODIGIANO** FERRARI LUIGI Via Madre Cabrini, 44 • **BARLASSINA** F.LLI GALIMBERTI Via Nazionale dei Giovi, 28/36 • **BOVISIO MASCIAGO** R & C C.so Milano, 118 •**CINISELLO BALSAMO** GBC V.le Matteotti, 66 • **MILANO** GBC Via Petrella, 6 • **MILANO** GBC Via Cantoni, 7 •**COLOGNO MONZESE** CASA DELLA MUSICA Via Indipendenza, 21 • **CORBETTA** PENATI Via Verdi, 28/30 • **CORSICO** EPM System V.le Italia,12 • **DESIO** P.GIORGIO OSTELLARI Via Milano, 300 •**LEGNANO** CENTROCOMPUTER PANDOLFI Via Corridoni,18 •**LISSONE** COMPUTEAM Via Vecellio, 41 • **LODI** FUTURA Via Solferino, 31 •**LODI** M.B.M. C.so Roma, 112 •**MELEGNANO** L'AMICO DEL COMPUTER SAS V.le Lombardia, 17 • **MONZA** BIT 84 Via Italia, 4 • **NOVATE MILANESE** IL CURSORE Via Cavour, 35 • **OPERA** I.C.O. Via dei Tigli, 14 • **SESTO SAN GIOVANNI** NIWA HARD & SOFT Via Bruno Buozzi, 94 • **VIMODRONE BERGAMO** IL COMPUTER SERVICE SHOP Via Padana Superiore, 197 • **BERGAMO** COMIF Via Autolinee, 10 • CORDANI Via dei Caniana, 8 • D.R.B. Via Borgo Palazzo, 65 • NEW SYSTEMS POINT Via Paglia, 36 • **ALZANO LOMBARDO** BERTULEZZI GIOVANNI Via Fantoni, 48 •**CARVICO** COMPUTER TEAM hi-tec Via Verdi, 1/B •**LOVERE** OTTICO OPTOMETRISTA ROVETTA P.zza Garibaldi, 6 • **SAN PELLEGRINO TERME** A.I.S. INTERNATIONAL Via S. Carlo, 25 •**SARNICO** COMPUTER POINT Via Lantieri, 52 •**BRESCIA** SISTHEMA Via Roma, 45 •**URGNANO BRESCIA E PROVINCIA** A.B.INFORMATICA Strada Statale Cremasca, 66 •**BRESCIA** COMPUTER CENTER Via Cipro, 62 • INFORMATICA 2000 Via Stazione, 16/B •MASTER INFORMATICA Via F.lli Ugoni, 10/8 •VIGASIO MARIO Portici Zanardelli, 3 •**BRENO** MISTER BIT Via Mazzini, 70 • **CASTREZZATO** CAVALLI PIETRO Via 10 Giornate, 14B •**CHIARI** (BS) ViETTI GIUSEPPE Via Milano, 1/B •**DESENZANO DEL GARDA** MEGABYTE P.zza Maluezzi, 1 •**GHEDI** DITTA BARESI RINO & C. Via XX Settembre, 7 •**GRATACASOLO** INFO CAM Via Provinciale, 3 •**COMO** IL COMPUTER Via Indipendenza, 90 •2M ELETTRONICA Via Sacco, 3 • **BARZANO'** ELTRONGROS VIA L. Da Vinci, 54 •**CASSAGO BRIANZA** EGA Via Mazzini, 42 • **ERBA** DATA FOUND computer shop Via A. Volta, 4 •**LECCO** CIMA ELETTRONICA Via Leonardo Da Vinci, 7 •FUMAGALLI Via Cairoli, 48 • **OLGIATE COMASCO** RIGHI ELETTRONICA Via G. Leopardi, 26 •**CREMONA** MONDO COMPUTER Via Giuseppina, 11/B • PRISMA Via Buoso da Novara, 8 • TELCO P.zza Marconi, 2/A • **CREMA** ELCOM/GBC Via VI Novembre, 56/58 • EUROELETTRONICA Via XX Settembre, 92/A • **MANTOVA E PROVINCIA** COMPUTER SAS Galleria, 7 •32 BIT (Computer Studio) Via Cesare Battisti, 14 • ELETTRONICA di BASSO V.le Risorgimento, 69 •**PAVIA** POLIWARE SRL C.so Carlo Alberto, 76 • **VIGEVANO** LOGICA INFORMATICA V.le Monte Grappa, 34 •M. VISENTIN C.so Vittorio Emanuele, 76 •**SONDRIO** CIPOLLA MAURO Via Tremogge, 25 • **SAN PIETRO DI BERBENNO** FOTONOVA V.Valeriana, 1 • **VARESE** DIMECO SISTEMI Via Garibaldi •IL CENTRO ELETTRONICO Via Morazzone, 2 • SUPERGAMES Via Carrobbio, 13 •**BUSTO ARSIZIO** BIT Via Gavinana, 17 • MASTER PIX SNC Via S. Michele, 3 •**CASTELLANZA** CRESPI GIUSEPPE & C. V.le Lombardia, 59 •**GALLARATE** PUNTO UFFICIO Via Raffaello Sanzio, 8 •GEMONIO SIDALCO Via Caprera, 10 •**GERENZANO** (VA) GRANDI MAGAZZINI BOSSI Via Clerici, 196 •**SESTO CALENDE** (VA) J.A.C. nuove tecnologie Via Matteotti, 38

PIEMONTE

ALESSANDRIA BIT MICRO Via Mazzini, 102 •SERVIZI INFORMATICI Via Alessandro III, 53 •**TORTONA** S.G.E. ELETTRONICA Via Bandello, 19 •**ASTI** RECORD C.so Alfieri, 166/3 • **ASTI CUNEO** ROSSI COMPUTERS C.so Nizza, 42 •**CUNEO** STUDIO SOFTWARE C.so Nizza, 4 •PUNTO BIT C.so Langhe, 26/C •ALBA SDI Via Vittorio Emanuele, 250 BRA • **FOSSANO** ASCHIERI GIANFRANCO C.so Emanuele Filiberto, 6 • BOSETTI Via Roma, 149 • **NOVARA** ELCOM SRL C.so Mazzini, 11 •PROGRAMMA 3 V.le Buonarroti,8 •PUNTO VIDEO C.so Risorgimento, 39/8 • **ARONA** COMPUTER Via Monte Zeda, 4 • **BORGOMANERO** ALL COMPUTER C.so Garibaldi,106 • **DOMODOSSOLA** MICROLOGIC Via Giovanni III, 2 • INTRA ELLIOTT COMPUTER SHOP Via Don Minzoni, 32 • **TORINO** ABA ELETTRONICA Via C. Fossati, 5/P • ALEX COMPUTER E GIOCHI C.so Francia, 333/4 • **VILLASTELLONE** CDM ELETTRONICA Via Marocchetti, 17 spedizione: C.so Savona, 43 • **TORINO** COMPUTER HOME SNC Via San Donato, 46/D • COMPUTING NEW Via Marco Polo, 40/E • DE BUG C.so Vittorio Emanuele II, 22 • DESME UNIVERSAL Via San Secondo, 95 • F.D.S. ALTERIO Via Borgaro, 86/D • IL COMPUTER Via Nicola Fabrizi, 140 bis • INFORMATICA ITALIA C.so Re Umberto, 129 (128) • MT INFORMATICA C.so Giulio Cesare, 58 • MUSIC 'S SHOP C.so Potenza, 177 • NEWBUSINESS COMPUTER Via Nizza, 45/F • PLAYGAMES SHOP Via Carlo Alberto, 39/E • RADIO TV MIRAFIORI C.so Unione Sovietica, 381 • SMT ELETTRONICA Via Bibiana, 83/B • TELERITZ C.so Traiano, 34 • **CHIERI** PAUL E CHICO VIDEOSOUND Via Vittorio Emanuele, 52 • **CIRIE'** BIT INFORMATICA Via Vittorio Emanuele, 154 • **COLLEGNO** HI-FI CLUB C.so Francia, 92C • **FAVRIA** MISTER PERSONAL Via Cattaneo, 52 • **IVREA** C.S.S. Stradale Torino N. 73 • MONCALIERI BAS C.so Roma, 47 •**PINEROLO** CERUTTI MAURO C.so Torino, 234 • **RIVAROLO** EUREX C.so Indipendenza, 5 •**RIVOLI** C.SE DIAM INFORMATICA C.so Francia, 1 46/bis • FULLINFORMATICA Via Vittorio Veneto, 25 • **SAN MAURO TORINESE** PANORAMA TORINO SPA Strada Settimo, 371 • **SETTIMO TORINESE** GAMMA COMPUTER Via Cavour, 3 A/B •**VERCELLI E PROVINCIA** DITTA ELETTROGAMMA C.so Bormida, 27 • ELETTRONICA Strada Torino, 15 • **BIELLA** C.S.I. TEOREMA Via Losana, 9 • SIGEST SRL Via Bertodano, 8 • **BORGOSESIA** REMONDINO FRANCO Via Roma, 5 • **COSSATO** FOTOSTUDIO TREVISAN Via XXV Aprile, 24/B • **TRINO** STUDIO FOTOGRAFICO IMARISIO P.zza Martiri Libertà, 7

VENETO

BELLUNO UP TO DATE di VIEL RENZO Via Vittorio Veneto, 43 • **PADOVA E PROVINCIA** GUERRA COMPUTERS FELTRE Via Mazzini, 10/C • **PADOVA** BIT SHOP Via Cairoli, 11 • COMPUMANIA Riviera Tisa da Camposampiero, 37 • COMPUTER POINT Via Roma, 63 • D.P.R. Vicolo Lombardo,4 •GIANFRANCO MARCATO Via Madonna della salute, 51/53 spedizione: Via Bergamini, 4 • SARTO COMPUTER Via Armistizio, 79 • ZELLA ADELIO P.zza De Gasperi, 31/A • **CITTADELLA ROVIGO** COMPUTER SERVICE Borgo Treviso, 150 • **ROVIGO TREVISO** CLINICA DEL RASOIO E DEL COMPUTER Via Fiume, 31/33 • **TREVISO** BIT 2000 Via Brandolini D'Adda, 14 • GUERRA EGIDIO &C. V.le Cairoli, 95 • **CONEGLIANO** DE MARIN COMPUTERS Via XX Settembre, 74 • **MONTEBELLUNA** SIDESTREET Via Salvo D'Acquisto, 8 • **PREGANZIOL** FALCONE ELETTROAUDIOVIDEO Via Terraggio, 116 • **VENEZIA E PROVINCIA** TELERADIO Fuga San Marco spedizione: IVANO BERTOLA Via Rossi Rubano • **MESTRE-VENEZIA** TREKILOWATT Via Torre Belfredo, 47 •**CHIOGGIA** T.C.H. Riva Vena, 889 • **SAN DONA' DI PIAVE** (VE) GUERRA COMPUTER Via Vizzotto,29 • REBEL Via F. Crispi, 10 • **SOTTOMARINA** TELFERT di Ferretto Flavio Via Chiesa • **SPINEA** RADIOCESTARO Via Roma, 89 • **VERONA** CASA DELLA RADIO Via Cairoli 10 • TELESAT Via Vasco De Gama ,8 spedizione: Ivano Bertola Via Rossi • **LEGNANO** RUBANO FERRARIN Via dei Massari,10 spedizione: Ivano Bertola Via Rossi • **VICENZA** ELETTRONICA BISELLO V.le Trieste, 427/429 • SCALCHI MARKET Via Cà Balbi, 139 • **CECCATO** GUERRA COMPUTER Via Dell'industrie Alte • **CAVAZZALE** SCHIAVOTTO Via Zanella, 21 • COMPUTER B.COSTO Via del Costo,34

FRIULI VENEZIA GIULIA

GORIZIA THIENE ELETTROCASA Via Roma, 67 • **GORIZIA PORDENONE** E.C.O. ELETTRONICA COMMERCIALE Via F.lli Cossar, 23 • **PORDENONE** RIGO V.le Cossetti, 5 • BRUNO DA PIEVE & C. Via Colombera,l7 • **VILLANOVA DI PRATA TRIESTE** PORCIA MDT P.zza Repubblica, 5 • **TRIESTE** AVANZO GIACOMO P.zza Cavana, 7 • COMPUTER SHOP Via P. Reti • **TRIESTE** COMPUTIGI Via XX. Settembre, 51 • **TRIESTE UDINE** CT Via Pascoli, 4 • **UDINE** MOFERT V.le Europa Unita, 41 spedizione:Mifert,2 Via Leopardi 2 • R.T. SISTEM UDINE Via L. Da Vinci, 99 • **MARTIGNACCO** INDRENO MATTIUSSI & C. Via Liciniana, 58

TRENTINO ALTO ADIGE

BOLZANO COMPUTER POINT Via Roma, 82A • **BOLZANO** MATTEUCCI PRESTIGE Via Museo, 54 spedizione: ELETTRON MATTEUCCI Via Parma • **BRUNICO** RADIO MAIR-ELECTRO Via Centrale 70 • **MERANO** ELECTRO RADIO HENDRICH Via Delle Corse, 106 • **SILANDRO** (BZ) **TRENTO** ELECTRO TAPPEINER P.zza Principale, 90 • **TRENTO** CRONST SAS Via G. Galilei, 25 • ELETTROCASA V.le Verona, 9

LIGURIA

GENOVA ABM COMPUTER P.zza De Ferrari, 24 rosso • **GENOVA SESTRI PONENTE** CENTRO ELETTRONICO Via Chiaravagna, 10R • **GENOVA** COMMERCIALE SOTTORIPA Via Sottoripa, 115/117 • **SAMPIERDARENA GENOVA** COMPUTER VIA D.G. Storace 4/rosso • **GENOVA** FOTOMONDIAL Via Del Campo 3-5-9-11-13 r • LA NASCENTE Via San Luca, 4/1 • **GENOVA IMPERIA** RAPPR-EL Via Borgoratti, 23/R • **IMPERIA CASTELLINO** Via Belgrano, 44 spedizione: SASA COMPUTER • **SANREMO** (Terso) CENTROHI-FI VIDEO Via della Repubblica, 38 • **VENTIMIGLIA CASTELLINO** Via Genova, 48 spedizione: SASA COMPUTER (Terso) • **LA SPEZIA E PROVINCIA** I.L. ELETTRONICA Via Vittorio Veneto, 123 • **FORNOLA DI VEZZANO SAVONA** I.L. ELETTRONICA Via Aurelia, 299 • **SAVONA** ATHENA INFORMATICA Via Carissimo e Crotti, 16/R • **CASTELLINO** C.so Tardy e Benech, 101 spedizione: SASA COMPUTER (Terso)

EMILIA ROMAGNA

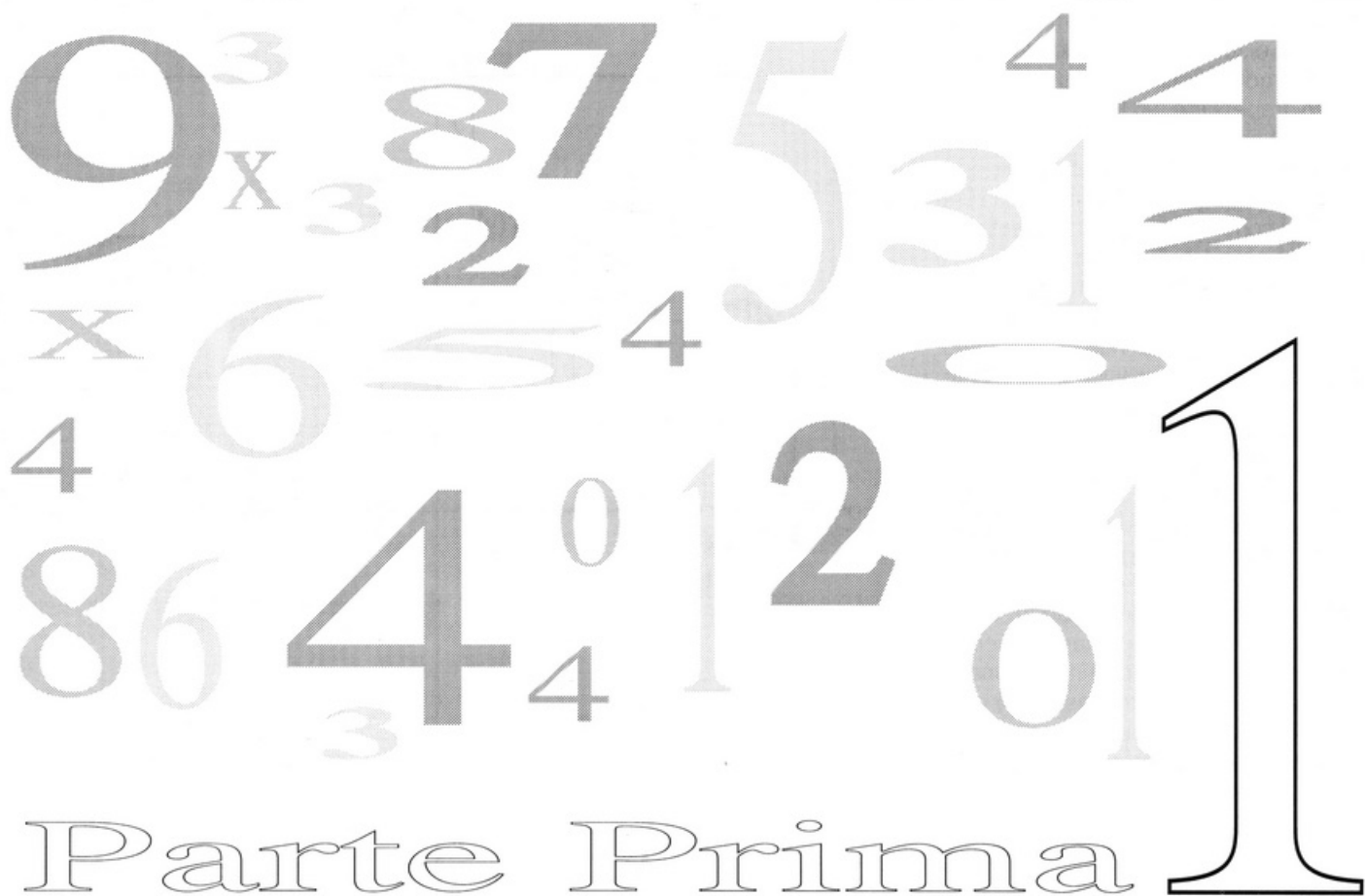
PIACENZA COMPUTER LINE Via G. Carducci, 4 • DELTA COMPUTER Via Martiri della Resistenza, 15/G • SOVER Via VI Novembre, 60

TOSCANA

AREZZO DELTA SYSTEM Via Piave, 13 • **FIRENZE** ATEMA Via Benedetto Marcello, 1a-1b • ELETTRONICA CENTOSTELLE Via Cento Stelle, 5/a-b • HELP COMPUTER Via degli Artisti, 15-A • PUNTO SOFT Via Vagnetti, 17 • TELEINFORMATICA TOSCANA Via Bronzino, 36 • **EMPOLI** WAR GAMES SDF Via Raffaello Sanzio, 126/A • **FIGLINE VALDARDO** (FI) NEW E.V.M COMPUTER Via Degli Innocenti, 2 • **PONTASSIEVE** CENTRO INFORMATICA Via Znojmo, 41 • **PRATO** (FI) COSCI F.lli Via Roma, 26 • **GROSSETO** COMPUTER SERVICE Via Dell'Unione, 7 • **LIVORNO** ETA BETA VIA San Francesco, 30 • FUTURA 2 Via Cambini, 19 • **PIOMBINO** ELETTRONICA ALESSI Via Cimarosa, 1 • **LIDO DI CAMAIORE** IL COMPUTER V.le Colombo, 21 6 • **S. ROMANO GARFAGNANA** (LU) SANTI VITTORIO Via Roma, 23 • **MASSA** EURO COMPUTER P.zza B. Bertagnini, 4 • FIRMWARE Galleria Pregliasco, 17 • **CARRARA** (MS) RADIO LUCONI Via Roma, 24/B • **PISA** C.H.S. SNC Via Carlo Cattaneo, 90/92 • ELECTRONIC SERVICE Via Della Vecchia Tranvia, 10 • IT - LAB Via Marche, 8A-8B • ditta TONY HI-FI Via G. Carducci, ang. Canto Dei Nicchio, 2 • **PISTOIA E PROVINCIA** ELECTRONIC SHOP Via Della Madonna, 49 • OFFICE DATA SERVICE Galleria Nazionale, 22 • **MONTECATINI TERME** ZANNI & C. C.so Roma, 45 TEL. 0572-79649 spedizione: Via Forini,l0 • **SIENA** VIDEO MOVIE Via Garibaldi, 17 • **CHIANCIANO TERME** ELECTRIC SHOP Via A. Casini, 51 • **MONTEPULCIANO** ELETTRONICA Via di Gracciano nel Corso, 111

Segue a pag. 98

INTEGRAZIONE NUMERICA

Parte Prima 1

Di Giovanni Michelon e Luigi Manzo

Affrontiamo in questo numero e nel successivo il problema del calcolo di un integrale definito per via numerica.

E' questo uno degli argomenti cardine dell'analisi numerica; capita infatti spesso, nella soluzione di vari problemi, di imbattersi in un integrale.

Cos'è un integrale

Dedichiamo questo paragrafo a tutti i lettori che si sono posti tale angosciante quesito.

Tanto tempo fa (no, non è una favola) i matematici greci cercarono una strategia generale per risolvere il "problema delle aree": calcolare l'area di un rettangolo o di un triangolo non era certo difficile, però l'area del cerchio creava già dei bei grat-

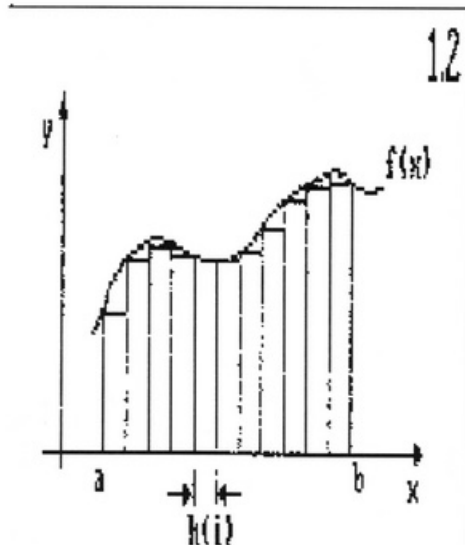
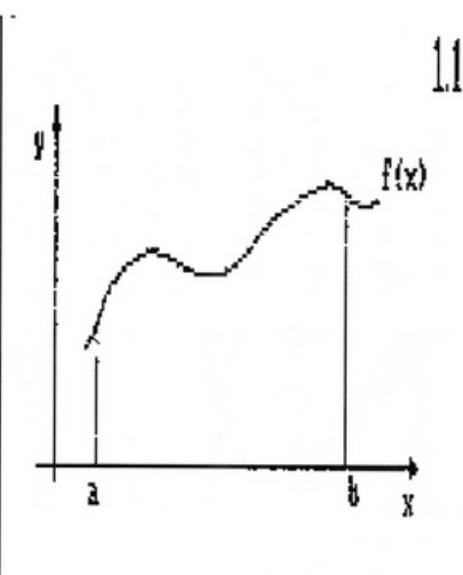
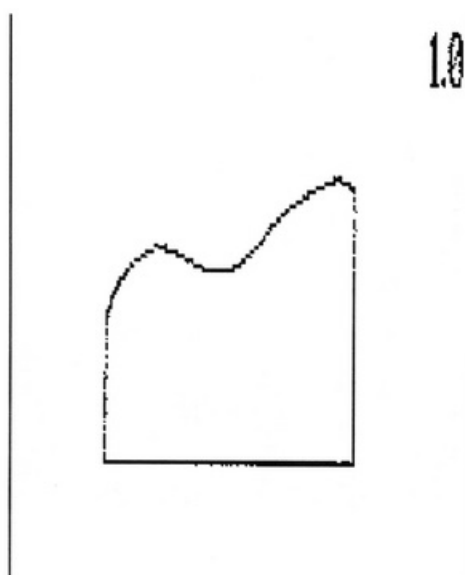
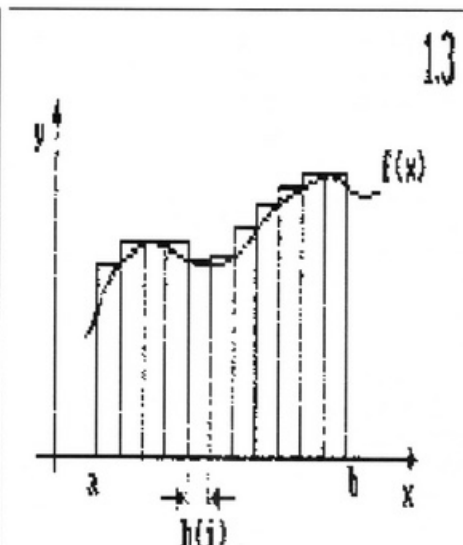
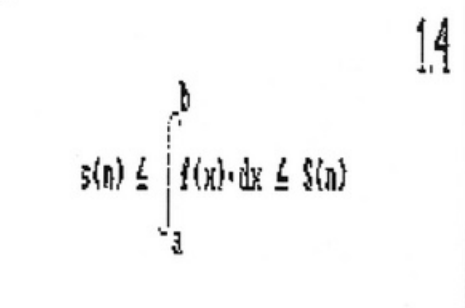
tacapi. Gli antichi greci non si scoraggiarono di certo: si resero ben presto conto che una circonferenza poteva essere approssimata con un poligono; anzi maggiore era il numero dei latitanto meglio il poligono si confondeva con una circonferenza. A questo punto, come allora quei matematici, vi sarete resi conto che il calcolo di un' area delimitata in tutto o in parte da curve non era più una difficoltà concettuale ma piuttosto materiale. Il problema delle aree fu allora anche chiamato problema delle quadrature poichè la sua soluzione consisteva proprio nel ridurre a quadrati, rettangoli o triangoli figure piane curvilinee.

Fermi là! non buttatevi subito a calcolare aree di cerchi: vediamo prima come impostare il discorso in via generale. Iniziamo per gradi: in figura 1.0 abbiamo un cosidetto trapezoide, un trapezio rettangolo, cioè, il cui lato obliquo è sostituito da una curva. Possiamo tranquillamente supporre che tale curva sia esprimibile da una certa funzione f(x), il tutto si presenta ora come in figura 1.1. Bene, affettiamo ora il nostro trapezoide come in figure 1.2 e 1.3,; al posto di ogni fetta possiamo prendere il rettangolino inscritto o circoscritto, del resto tanto più sottili saranno le fette, tanto più piccola sarà la differenza fra i due rettangoli.

A questo punto possiamo dare anche la definizione rigorosa di integrale definito di una funzione f(x); torneremo in seguito sul significato dell'aggettivo "definito". Sempre riferendoci alle figure 1.2 e 1.3, consideriamo le due sommatorie, quella "S(n)" di tutte le aree dei rettangolini circoscritti e quella "s(n)" di quelli inscritti (max(i) e min(i) sono le ordinate massima e minima della funzione nel singolo rettangolo); possiamo ora considerare tutte le possibili suddivisioni dell'intervallo [a,b], avremo sempre che S(n) è maggiore o uguale ad s(m) (n ed m indicano due suddivisioni diverse).

Bene, la virtù sta nel mezzo dicevano gli antichi, ed anche l'integrale della f(x); cioè scegliendo fra tutte le possibili suddivisioni si otterranno valori di S(n) ed s(m) sempre più vicini, possiamo perciò dire che l'integrale della f(x) sia il valore che separa S(n) da s(m).

In figura 1.4 si vede il simbolo di integrale, a e b sono detti estremi di integrazione, f(x)*dx rappresenta l'area di un ret-

figura 1
Definizione di integrale

tangolino di larghezza infinitesima, il simbolo di integrale in effetti altro non è se non una lettera "S" (per somma), che ha subito la tortura medioevale del cavalletto.

Abbiamo quindi un metodo pratico per calcolare un'area, prima di continuare a sviluppare il concetto di integrale facciamo una piccola parentesi ludica: calcoliamo l'area del cerchio.

Banale dirà qualcuno, tutti sanno che è PI*R^2 (PI=3,14...), però, se volessimo avere una precisione maggiore, (anche di quella della vostra calcolatrice) dobbiamo fare proprio come gli antichi greci, considerare cioè poligoni con un numero sempre maggiore di lati. Noi proponiamo il programma PI che adotta, appunto, questa tecnica per ottenere le cifre di PI, calcolando cioè l'area di un cerchio di raggio unitario, (naturalmente siamo ben al di sotto del record mondiale di più di cento milioni di cifre decimali, ottenuto nel 1987 con ben altre tecniche).

Ritorniamo a discorsi più impegnati: finora abbiamo una nozione teorica ed un metodo approssimativo per valutare un integrale; questo non poteva naturalmente soddisfare i matematici, che cercavano delle formule esatte; a quell'epoca i greci riuscirono a trovare con complicate ed ammirevoli costruzioni geometriche formule esatte soltanto per alcune curve.

Per proseguire arriviamo al Seicento; i matematici di quell'epoca stavano cercando una soluzione al problema della primitiva (si suppone nel seguito che sappiate cosa sia una derivata in modo quantomeno intuitivo, così come è stata descritta nel terzo numero di questa rivista).

In termini matematici il problema era, data una funzione f(x), quello di trovare la, o le funzioni, la cui derivata coincidesse con la f(x), di trovare insomma le "primitive" della f(x). Torricelli riuscì a dimostrare che le primitive della f(x) si possono ottenere con la formula di figura 2.0: cioè la primitiva è data dall'area del trapezoide con base l'intervallo [x0,x] più una costante qualunque.

Chiariamo ora questo discorso riprendendo l'esempio della legge oraria del moto, quella legge che dà la posizione di un oggetto in funzione del tempo; avevamo visto come la derivata rappresentasse la legge oraria delle velocità (cioè velocità dell'oggetto in funzione del tempo). I matematici del Seicento volevano, data la

Fig. 2

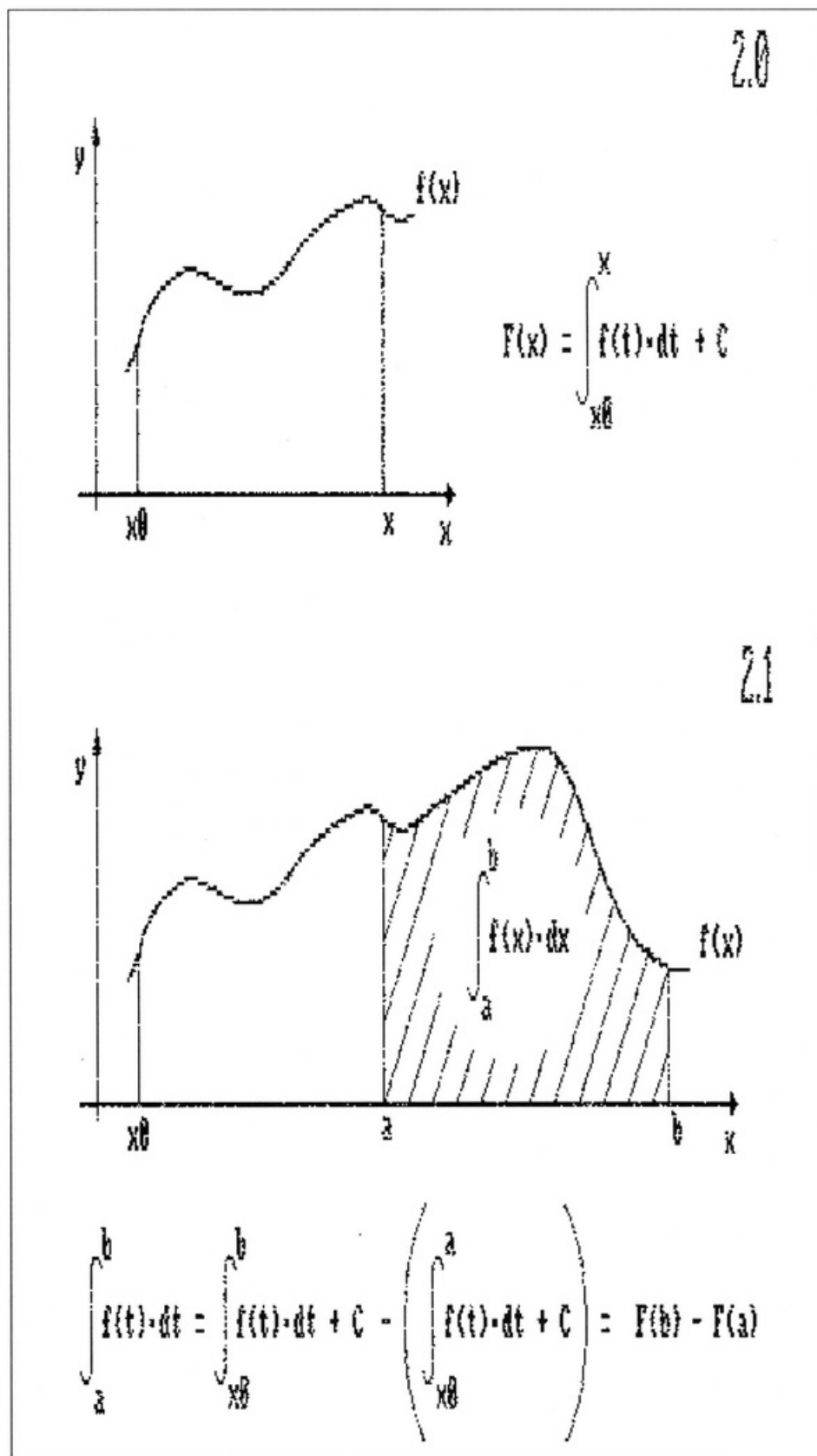
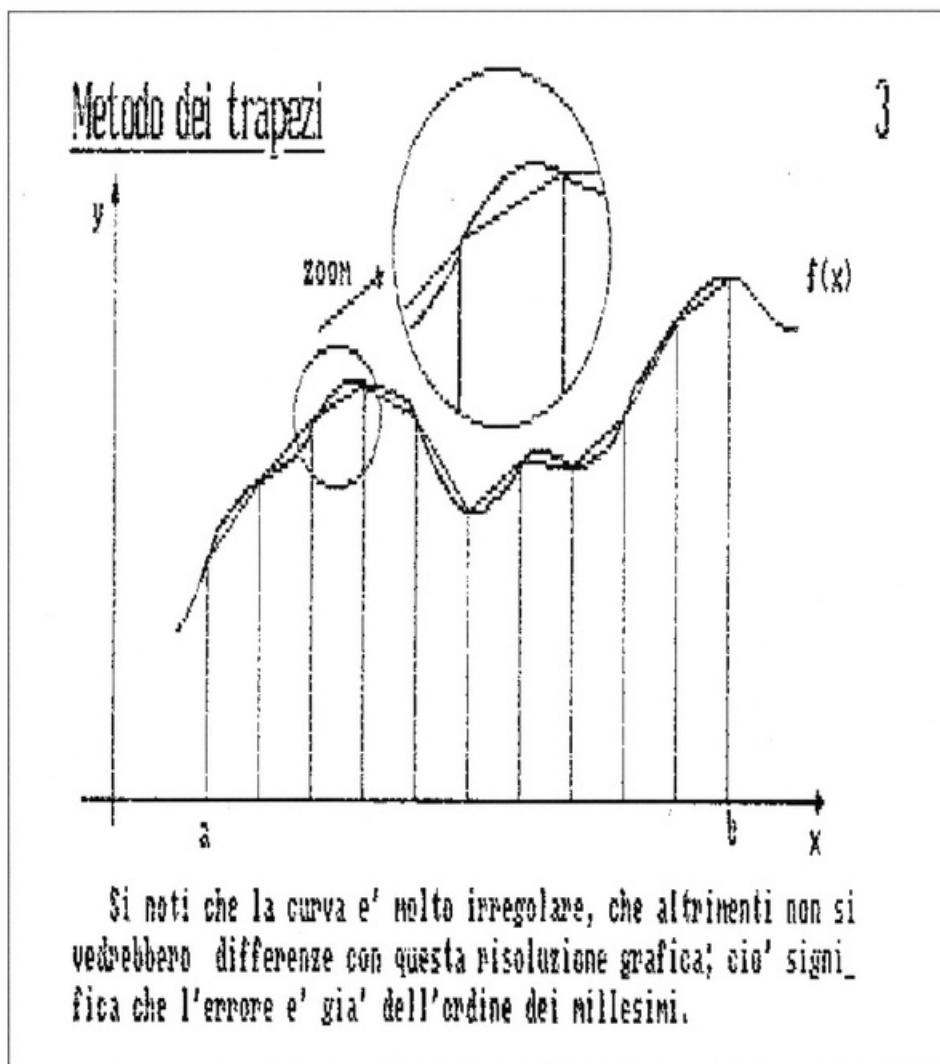



Fig. 3

legge oraria delle velocità, trovare la legge oraria del moto.

Non è questa la sede per dimostrare il teorema di Torricelli, però pensando alle leggi orarie non dovrebbe esservi difficile comprendere da dove spunti la costante C; supponiamo di prendere la bicicletta e di avere come legge oraria la seguente: "avanti per 10 minuti a 20 km/h, poi a destra per 5 minuti a 30 km/h": chiunque saprebbe dire la posizione dopo 15 minuti, purchè venga specificato il punto di partenza! C rappresenta tale punto di partenza, è perciò chiaro perché il problema della primitiva ha infinite soluzioni simili. Se ora osservate la figura 2.1 potete notare che l'integrale di f(x) sull'intervallo [a,b] si può esprimere facilmente come differenza fra il valore della primitiva F(x) in b e quello in a, indipendentemente dal valore della costante arbitraria C.

In altre parole i matematici del Seicento trovarono un modo per ottenere le formule per le aree: basta infatti essere in grado di trovare la F(x), la primitiva cioè della f(x).

Bella forza dirà qualcuno, ma la F(x) chi ce la dà? Beh, in generale non ve la dà proprio nessuno, in alcuni casi però le primitive si conoscono, e naturalmente il caso più semplice è quello dei polinomi (ancora loro?!).

Cosa ci fanno qui i polinomi? State a vedere cosa pensarono di fare i signori,Newton e Cotes.

Il metodo di Newton - Cotes: i trapezi

Nelle figure 1.2 e 1.3 abbiamo visto due tecniche per calcolare un'area; come però già si vede in quelle figure i rettangoli sono un "po' restii" a seguire profili curvilinei, una ovvia osservazione suggerisce di usare dei trapezi, anzichè dei rettangoli (vedi figura 3). E' questo il metodo dei trapezi: è il nonno dei metodi di integrazione numerica, e nonostante l'età è tuttaltro che prossimo al pensionamento, infatti, usando il programma IntNum, potete verificare che i risultati, sia in termini di tempo di calcolo che di precisione, non sono poi da gettare.

Ogni nonno che si rispetti ha una bella famigliola, facciamone dunque la conoscenza!

I signori Newton e Cotes pensarono: "il metodo dei trapezi consiste nel calcolare l'area sotto una retta (il lato obliquo del trapezio), ora una retta è rappresentata da un polinomio di primo grado, se usassimo una parabola (polinomio di secondo grado) protremmo ottenere dei risultati migliori!"

Nell'intento di ottenere risultati sempre migliori si arrivò a quella serie di metodi noti come formule di quadratura di Newton-Cotes.

Chiariamo meglio: data la funzione f(x) e l'intervallo [a,b], suddividiamo [a,b] in n parti uguali, diciamo tratto ognuna di queste parti: in parole povere le altezze dei trapezi sono tutte uguali. Ciò che vogliamo fare è approssimare la f(x) con una retta oppure con un polinomio di grado superiore. Come avrete capito al volo, se siete nostri fedeli lettori, siamo di fronte ad un problema di interpolazione; questa volta utilizzeremo un nuovo metodo di interpolazione dovuto a Lagrange, è questo un metodo non conveniente dal punto di vista computazionale (richiede molti calcoli) ma comodissimo dal punto di vista teorico, infatti, come vedremo, ci permetterà di ottenere delle formule estremamente semplici e compatte. Facciamo perciò una digressione e vediamo cosa ci propone il signor Lagrange.

Interpolazione alla Lagrange

Supponiamo di avere una funzione f(x) e di volerla interpolare con una retta per i nodi x0 e x1, cerchiamo un modo semplice per scrivere l'equazione della retta. Osserviamo l'espressione di figura 4.0): è senza ombra di dubbio l'equazione di una

retta 4.0

$$p(x)=f(x_0)\frac{(x-x_1)}{(x_0-x_1)}+f(x_1)\frac{(x-x_0)}{(x_1-x_0)}$$

parabola 4.1

$$p(x)=f(x_0)\frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)}+f(x_1)\frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)}+f(x_2)\frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)}$$

Polinomi di Lagrange 4.2

$$p(x):=\sum_{i=0}^{n} l(x,i)*f(x_i) \qquad l(x,i):=\prod_{j=0,\,j<>i}^{n}\frac{(x-x_j)}{(x_i-x_j)}$$

Fig. 4

retta (nella variabile x, le altre sono costanti); se x=x0 allora otteniamo:

p(x0)=f(x0)*1+f(x1)*0

se invece x=x1 allora abbiamo:

p(x1)=f(x0)*0+f(x1)*1

insomma p(x) soddisfa alle condizioni di interpolazione poichè assume nei due nodi x0 e x1 gli stessi valori della f(x). Beh, ormai il trucco è svelato, per ottenere una parabola sui nodi x0, x1 e x2 si userà la formula di figura 4.1); anche qui è istantaneo verificare che p(x) e f(x) coincidono sui nodi.

Osserviamo, sempre in figura 4.1), che p(x) si scrive come somma di prodotti del valore di f(x) su un nodo per un polinomio di secondo grado fatto in modo particolare. Questo polinomio, che si indica con l(x,i) gode della proprietà di essere nullo in tutti i nodi diversi da quello i-esimo, e di valere 1 in quest'ultimo. Notate ancora che l(x,i) è dato dal prodotto di due termini, mentre i nodi sono tre, manca infatti il termine relativo al nodo i-esimo, proprio perché lì il polinomio deve essere non nullo.

Possiamo quindi scrivere il caso generale per un polinomio di grado n come somma del prodotto di f(xn(i)) (con xn(i) intendiamo, al solito, il nodo i-esimo) per il polinomio l(x,i); in figura 4.2) è riportata l'espressione generale, si noti che l(x,i) è espresso, essendo il prodotto di più fattori col simbolo di produttoria (p greca maiuscola), di significato analogo a quello di sommatoria.

Newton-Cotes: caso generale

Bene, un ultimo sforzo per ottenere le formule di Newton-Cotes per diversi gradi polinomiali.
Al solito abbiamo funzione ed intervallo di integrazione [a,b], sia m il numero dei tratti e h=(b-a)/m l'ampiezza di ogni tratto, sia n il grado del polinomio che utilizzeremo su ogni tratto.

Con riferimento alla figura 5.0), calcoliamo l'integrale della f(x) fra il nodo xn(i) e xn(i+1) con il polinomio p(x); scriviamo p(x) alla Lagrange e otteniamo ciò che si vede in figura 5.1) dove con x(j) si intendono gli n+1 nodi fra xn(i) ed xn(i+1), dove il primo e l'ultimo degli x(j) coincide rispettivamente con xn(i) ed xn(i+1). Si dimostra con un piccolo trucchetto che vale anche la seconda ugaglianza (quella indicata dalla freccetta): i w(j) sono delle costanti indipendenti dalla f(x) e da [a,b], vengono detti "pesi" di Newton-Cotes.

La formula finale è infine quella di figura 5.2); i w(j) si dicono pesi poichè l'espressione della formula finale ricorda tanto quella di una "media pesata".

Ordine di convergenza

In figura 6) è riportata la tabella con i pesi di Newton-Cotes fino all'ordine 7. Il motivo di tale limitazione è che per ordini superiori compaiono pesi negativi che tendono ad indurre ad errori di cancellazione. La tabella è a metà, poichè i pesi della seconda metà sono "simmetrici" a quelli della prima. Per curiosità si sono riportati anche i nomi storici delle serie dei pesi.

Si dice che una formula di quadratura ha grado polinomiale m se è esatta per tutti i polinomi fino al grado m, cioè calcolando l'integrale di un polinomio di grado fino ad m con quella formula si ottiene il risultato esatto.

Per le formule di Newton-Cotes si ha un fatto interessante: se la formula è di ordine n ed n è pari, allora il grado polinomiale è n+1, altrimenti è n. Questo significa che le formule di ordine 2,4 o 6 regalano, in offerta omaggio, un grado in più ad un prezzo minore di quella di ordine successivo (3,5,7). Cioè, per intendersi, la regola di Simpson ha lo stesso grado della regola "pulcherrima", però vi "costa" solo tre valutazioni di funzione anzichè quattro.

Per confrontare fra loro le varie formule dovete imporre uno stesso metro di valutazione: supponiamo che abbiate scelto due formule di ordine n1 ed n2 allora dovrete farle lavorare, rispettivamente, su m1 e m2 tratti in modo che m1*n1 sia uguale a m2*n2; imponete così uno stes-

so numero di valutazioni di funzione, in pratica quindi lo stesso tempo di esecuzione, potete allora verificare quale delle due formule è più precisa.

Instancabili nel ripeterci, vi consigliamo di provare e confrontare i metodi proposti; noi proponiamo questo mese il programma IntNum V1.0i, "i" sta per incompleto, come preannunciato infatti, il discorso sugli integrali continuerà sul prossimo numero, dove presenteremo la versione definitiva.

Integrali con singolarità

Sfruttiamo l'argomento degli integrali per parlare dei problemi che le singolarità di una funzione possono causare in un programma. Innanzitutto diciamo che un valore x0 è punto di singolarità (o singolare) per una funzione f(x) se f(x) non è definita in x0. Vediamo con un esempio di chiarire il concetto: presa la funzione f(x)=1/(1-x*x) abbiamo che '-1' e '1' sono valori singolari.

In un programma Basic questo si risolve con un 'division by zero error', cosa che spesso è piuttosto spiacevole. Per evitare tali inconvenienti di solito si usano valori che si avvicinano a quello critico (0.99999 o simili).

Nel caso però si debbano fare più tentativi per ottenere un risultato accettabile, diventa conveniente disporre di un algoritmo che si porti vicino al punto critico, in modo che a noi resti solo da decidere il numero di passi di avvicinamento.

Siano dunque x0 il punto singolare e x1 un punto diverso da x0, in cui la f(x) sia definita; osservate ora il seguente programmino:

```
x:=x1
CICLO da 1 fino NumAvv
  x:=( (K-1)*x0 + x ) / K
FINE
```

Se K è maggiore di uno allora x si tenderà ad assumere valori sempre più prossimi ad x0, senza però mai raggiungerlo, a meno che NumAvv sia infinito. Supponiamo per esempio che K sia uguale a 2, allora alla prima iterazione x assumerà il valore medio fra x0 ed x1; al secondo giro x sarà a metà fra x0 e il valore precedente, cioè sarà ad un quarto della distanza fra x0 e x1: in parole povere con K=2 si di-

Fig. 5

5.0

$$\int_{xn(i)}^{xn(i+1)} f(x)\cdot dx \cong \int_{xn(i)}^{xn(i+1)} p(x)\cdot dx$$

5.1

$$\int_{xn(i)}^{xn(i+1)} p(x)\cdot dx = \sum_{j=0}^{n} f(x_j)\cdot \int_{xn(i)}^{xn(i+1)} l(x,j)\cdot dx = \sum_{j=0}^{n} f(x_j)\cdot h\cdot w_j$$

5.2

Formula generale di Newton-Côtes

$$\int_{a}^{b} f(x)\cdot dx \cong \sum_{i=0}^{N-1} h\cdot \sum_{j=0}^{n} f(x_j)\cdot w_j$$

figura 6 Tabella pesi Newton-Côtes

n	c1	c2	c3	c4	dn	Regola di/dei
1	1	-	-	-	2	trapezi
2	1	4	-	-	6	Simpson
3	1	3	-	-	8	3/8 o "pulcherrina"
4	7	32	12	-	90	Milne
5	19	75	50	-	288	-
6	41	216	27	272	840	Weddle
7	751	3577	1323	2989	17280	-

Fissato n w_i e' dato da : $w_i := n * c_i / dn$

Fig. 6

mezza la distanza ad ogni iterazione.

Nel programma IntNum abbiamo previsto la possibilità di punti singolari negli estremi di integrazione, e fissato K=5.

I programmi

Il primo programma di questo mese è PI, che, come preannunciato, serve per ottenere le prime cifre di PI. La versione che proponiamo utilizza l'aritmetica di macchina in doppia precisione. Il programma è scritto in Modula2; questo programmino ha essenzialmente uno scopo ludico, oltre a voler servire da esempio di soluzione del problema delle aree.

Passiamo ora ad IntNum, il programma che implementa le formule di Newton-Cotes; IntNum non è completo, lo sarà alla prossima puntata ("prossimamente sui vostri schermi"), quando parleremo dei miglioramenti possibili sulle formule fin qui viste.

Il programma è costruito in modo da essere il più compatto possibile e nello stesso tempo per rendere il più chiara possibile la traduzione in programma degli algoritmi. Questo va a scapito della velocità di esecuzione, che non abbiamo però preso in considerazione; IntNum serve infatti da banco di prova per testare pregi e difetti dei metodi proposti.

Veniamo ora alla struttura di IntNum. Diciamo subito dove va dichiarata la funzione da integrare, osservando il listato, subito sotto i commenti di presentazione, salta agli occhi la tipica DEF FN; per default f(x)=SIN(x), che integrata fra 0 e PI deve dare come risultato...beh, sta a voi scoprirlo!

I corpo principale del programma attende che da menù sia selezionato il metodo che interessa, fatto ciò richiede gli estremi di integrazione, l'ordine del metodo e il numero dei tratti. Alla domanda "Singolarità (a,b,N)?" rispondete con 'à o 'b' se uno dei due estremi è singolare, con 'n' se nessuno dei due lo è; se avete risposto 'à o 'b' allora il programma vi chiederà il numero dei passi di avvicinamento, che dovrete stabilire a seconda del tipo di funzione, e dell'ampiezza dei tratti.

Il programma procede eseguendo la subroutine 'calcola', questa, fissati gli estremi ed il passo di calcolo (pc) lancia la procedura che implementa il metodo stabilito.

Esaminamo dunque 'newtoncotes'; la prima cosa da fare è fissare i pesi a seconda dell'ordine (n) scelto, fatto ciò si passa ad un ciclo sui tratti. E' questo ciclo che fissati in 'eì ed 'es' gli estremi di ogni tratto usa 'intnc' per calcolare la formula di Newton-Cotes.

Infine si ritorna al modulo principale che stampa il risultato, contenuto nella variabile 's'.

L'analisi del risultato, come sempre in calcolo numerico, ha una importanza fondamentale, non ci rimane però qui lo spazio per affrontarla, cercate comunque di farvene un'idea provando IntNum. Potrete poi confrontare le vostre impressioni leggendo il prossimo numero, dove avrete la dimostrazione che con una corretta analisi si possono ottenere notevoli miglioramenti.

UMBRIA

PERUGIA LA MIGLIORATI Via S. Ercolano, 310 • **BASTIA UMBRA** COMPUTER STUDIO'S VIA IV Novembre, 18/A • **CITTA' DI CASTELLO** WARE Via Dei Casceri, 31 • **MATERA** GIOVANNI GAUDIANO ELECTRONICS Via Roma

PUGLIA

BARI ARTEL Via Guido D'Orso, 9 • COMPUTER'SARTS V.leb Meucci, 12/B • PAULICELLI SABINO & FIGLI Via Fanelli, 231/C • **BARLETTA** F. FAGGELLA C.so Garibaldi, 15 • GIANNI FAGGELLA P.zza D'Aragona, 62A • **CASTELLANA** LONUZZO GIUSEPPE Via Nizza, 21 • **BRINDISI** MARANGI E MICCOLI Via Prov. San Vito, 165 • **FRANCAVILLA FONTANA** MILONE GIOVANNI Via San Francesco D'Assisi, 219 • **FOGGIA** BOTTICELLI GUIDO Via San Pollice, 2 • E.C.I. COMPUTER Via Isonzo, 28 • LA TORRE SAS V.le Michelangelo, 185 • **SAN SEVERO** IL DISCOBOLO Via T. Solis, 15 • **LECCE** BIT Via 95° Reggimento Fanteria, 87/89 • TRICASECEDOK INFORMATICA Via Roma, 31 • **TARANTO** FERNANDO DE SANTIS Via Muro Maglie • ELETTROJOLLYCENTRO Via De Cesare, 13 • TEA Via Regina Elena, 101

CAMPANIA

ATRIPALDA FLIP FLOP Via Appia, 68 • **BENEVENTO** E.CO. INFORMATICA Via Pepicelli, 21/25 • **CASERTA** O.P.C. Via G.M. Bosco, 24 • MADDALONI M.P. COMPUTER Via Napoli, 30 • **NAPOLI** BABY TOYS Via Cisterna Dell'Olio, 5/bis • CASA MUSICALE RUGGIERO Int. Stazione Napoli • C.LE F.S. P.zza Garibaldi, 74 • CENTRO ELETTRONICO CAMPANO Via Epomeo, 121 • CI.AN Galleria Vanvitelli, 32 • CINE NAPOLI SNC Via Santa Lucia, 93/95 • DARVIN Calata San Marco, 26/25 • ELETTRONICA RO.DA.LO. Via Epomeo, 216/B • GIANCAR 2 P.zza Garibaldi, 37 • GRUPPO BUSCH Galleria Umberto, 55 • ODORINO L.go Lala, 22/A-B • R 2 SRL Via F. Cilea, 285 • TOP VIDEO - TOP COMPUTER VIA S. Anna Dei Lombardi, 12 • SPY Via San Giacomo Dei Capri, 36B/C • VIDEOFOTOMARKET Via S. Brigida, 19 • **S. ANASTASIA** (NA) SPADARO Via Romani, 93 • **CASORIA** ELECTRONIC DAY Via Delle Puglie, 17 • TUFANO S.S. Sannittica, 87 • **CASTELAMARE DI STABIA** SOF SUD V.le Europa, 59 • **FRATTAMAGGIORE** ELETTRONICA 2000 C.so Durante, 40 • **MUGNANO** GATEWAY Via Napoli, 68 • **PORTICI** NUOVA INFORMATICA SHOP Via Libertà, 185/191 • **POZZUOLI** BASIC COMPUTER C.so Garibaldi, 34 • **STRIANO** FALCO ELETTRONICA Via Sarno, 100 • **TORRE ANNUNZIATA** TECNOTRE Via P. Fusco, 1/F • **TORRE DEL GRECO** (NA) TECNO Via Vittorio Veneto, 28 • **SALERNO** COMPUTER MARKET C.so Vittorio Emanuele, 23 • **BATTIPAGLIA** (SA) KING COMPUTER Via Olevano, 56 • **EBOLI** (SA) DIMER POINT Via C. Rosselli, 20

CALABRIA

CATANZARO C. & G. COMPUTER Via F. Acri, 28 • PAONE SAVERIO & FIGLI Via F. Acri, 93/99 • **CROTONE** COMPUTER HOUSE Via Bologna (L.go Ospedale) • **VIBO VALENTIA** OTTICA FOTO NELLO RUELLO C.so Vittorio Emanuele, 177 • **COSENZA** SIRANGELO COMPUTER Via Parisio, 25 • HI-FI ALFANO GIUSEPPE Via Baldacchini, 109 • **CASTROVILLARI** AMANTIA ELIGIO ANNICCHIARICO &C. SAS Via Roma, 21 • **CORIGLIANO** SCALO ALFA COMPUTER Via Nazionale, 341/A • **LAMEZIA TERME** ING. FUSTO SALVATORE C.so Nicotera, 99 • **REGGIO CALABRIA** CONTROL SYSTEM Via San Francesco Da Paola, 49D/E • SYSTEM HOUSE Via Fiume ang. Palestino, 1 • **LOCRI** (RC) COMPUTER SHOP V.le Matteotti, 50-52 • **TAURIANOVA** PICIEFFE SNC C.so F. Sofia Alessio

ESIGI SEMPRE LA GARANZIA DELLA COMMODORE ITALIANA

AMIGA MAGAZINE N.6

SERVIZIO LETTORI **Compilare e spedire in busta chiusa a: GRUPPO EDITORIALE JACKSON Area Consumer - Via Rosellini, 12 - 20124 Milano**

A) Come giudichi questo numero di Amiga Magazine?

- ☐ Ottimo
- ☐ Molto Buono
- ☐ Buono
- ☐ Discreto
- ☐ Sufficiente
- ☐ Insufficiente

B) Quale(i) articolo(i) o rubrica hai apprezzato di più?

Quale meno?

C) Cosa ti piacerebbe leggere nei prossimi numeri di Amiga Magazine?

D) Ti è piaciuto il supergame?

E) Quante persone leggono la tua copia di Amiga Magazine?

F) Che computer possiedi?

Quale(i) computer intendi acquistare in futuro?

G) Leggi altre riviste Jackson?
☐ SI ☐ NO

Quali?

H) Leggi altre riviste dedicate al tuo computer?
☐ SI ☐ NO

Quali?

I) Oltre alle riviste dedicate al computer quali sono le tue letture preferite?

L) Quali sono i tuoi hobbies e maggiori interessi?

- ☐ Sport
- ☐ Musica
- ☐ Videoregistrazione
- ☐ Hi-Fi
- ☐ Fotografia
- ☐ Automobile
- ☐ Moto
- ☐ Viaggi

Altro

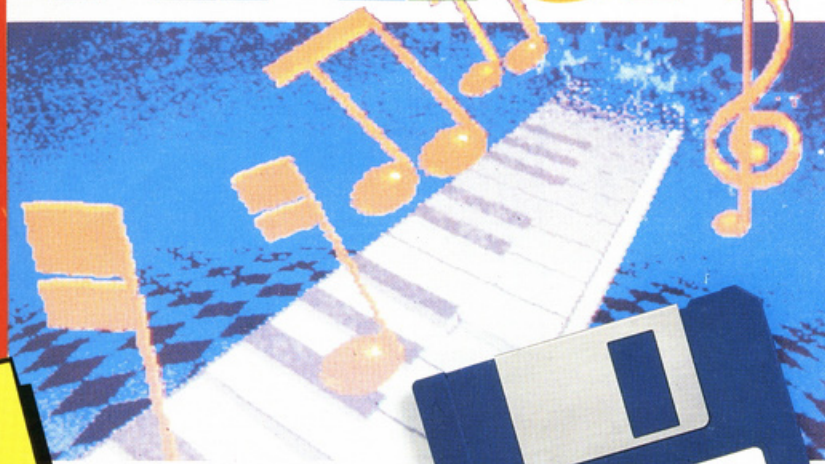
Nome

Cognome

Indirizzo

Età ____ **Professione**

Città

Prov. ____ **C.a.p.** ____ **Tel.**

BOMBER
TACTICAL AIR COMBAT
Veloce e divertente da giocare, Bomber ti regala una dettagliata e accurata combinazione di simulazione di volo, combattimento aria-aria e combattimento terra-aria, per il più completo gioco di simulazione mai realizzato.
• Realistico paesaggio in 3D
• Scegli fra 7 diversi aerei da combattimento
• Ricerca e seleziona le armi
• Unico sistema di controllo della visuale esterna che permette il movimento su tutti gli assi.
MANUALE INTEGRALE IN ITALIANO!
ACTIVISION
© ACTIVISION UK LTD 1989
ALL RIGHTS RESERVED
LEADER
DISTRIBUZIONE
Amiga screen shots shown
AUTO-ARM
CLEAR
END
INFO
Mig-27
Flogger
3D
INFO
SELECT
F4 Phantom
2D
INFO
SELECT